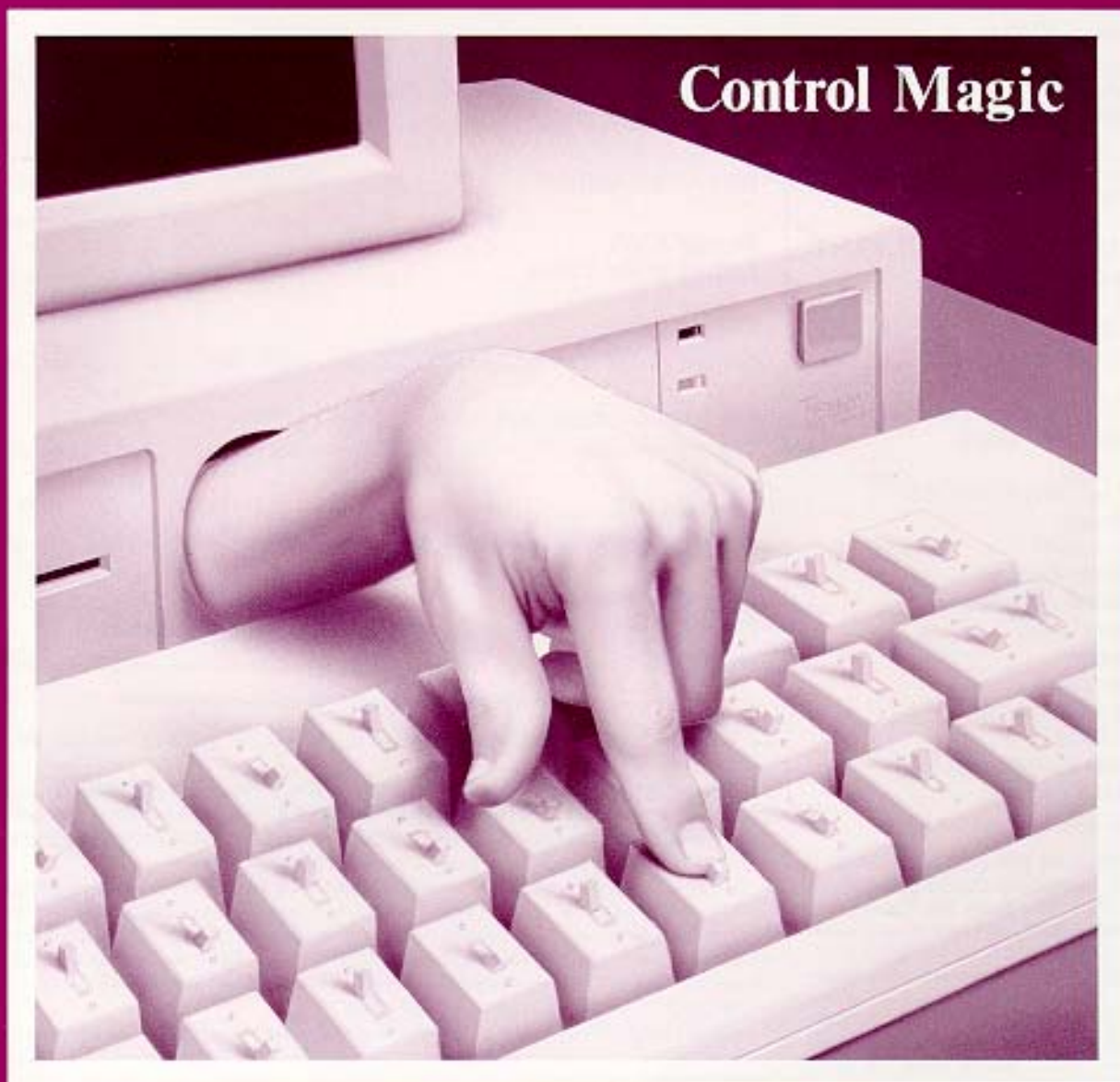


Circuit CELLAR LINK

MICROCOMPUTER APPLICATIONS





PUBLISHER

Stephen J. Walters

EDITORIAL DIRECTOR

Steve Ciarcia

EXECUTIVE EDITOR

Harv Weiner

TECHNICAL EDITORS

Kenneth Davidson

Jeff Bachiochi

CONTRIBUTING EDITORS

Thomas Cantrell

Edward Nisley

CIRCULATION DIRECTOR

Jeannette Do jan

CIRCULATION ASSISTANT

Diane Morey

CIRCULATION CONSULTANT

Gregory Spitzfaden

PRODUCTION MANAGER

Tricia Dziedzinski

BUSINESS MANAGER

Daniel Rodrigues

STAFF RESEARCHERS

Northeast

Eric Albert

William Curlew

Richard Sawyer

Robert Stek

Midwest

John Elson

Tim McDonough

West Coast

Frank Kuechmann

Mark Voorhees

Cover Illustration by Robert Tinney

CIRCUIT CELLAR INK (ISSN 0896-8985) is published bi-monthly by Circuit Cellar Incorporated, 4 Park St., Suite 20, Vernon, CT 06066 (203)875-2751. Second-class postage applied for a Vernon, CT and additional offices. One year (6 issues) charter subscription rate U.S.A. and possessions \$14.95. Canada \$17.95, all other countries \$26.95. All subscription orders payable in US funds only, via international postal money order or check drawn on US bank. Direct subscription orders to Circuit Cellar INK, Subscription, PO Box 3378, Wallingford, CT 06494 or call (203)875-2199.

TABLE of CONTENTS

EDITORIAL:

Control Magic by Steve Ciarcia 1

FEATURES:

Power-Line-Based Computer Control
The X-10 PL513 Power Line Interface Module
 by Ken Davidson 4

The Home Satellite Weather Center--Part 3:
Weather Databases and System Software Overview
 by Mark Voorhees 19

SOFTUART

Software Emulation of Full-Duplex Serial Channels
 by Bill Curlew 42

DEPARTMENTS:

Reader's Ink
Letters to the Editor 2

Visible Ink
Letters to the Circuit Cellar INK Research Staff 15

Ink Spot -- Guest Editorial
The Information Free-For-All by Mark Dahmke 27

ConnecTime
Excerpts from the Circuit Cellar BBS by Ken Davidson 28

Firmware Furnace
Video Signal Timing and Real-Time Interrupt Control
 by Ed Nirley 34

Circuit Cellar BBS - 24 Hrs. 300/1200/2400 bps, 8 Bits, No parity, 1 Stop Bit, 203-871-1988

The schemata 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-assembled projects, Circuit Cellar INK disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published in Circuit Cellar INK.

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

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept., PO Box 3378, Wallingford, CT 06494



EDITOR'S **I N K**

Control Magic

Computer control is not magic. As much as we'd like to think that there is this little guy in the computer who thinks through a problem and takes appropriate action independent of our concern, real knowledge about control interfacing and computer process-control hardware involves a determined intellectual focus and interest.

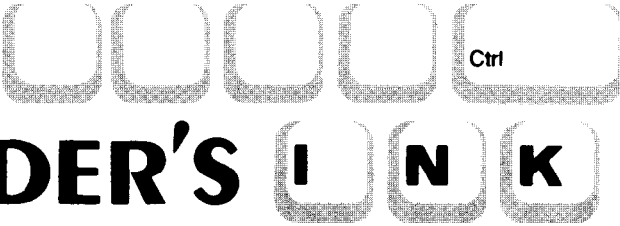
It will come as little surprise to most of you that computer control is one of my favorite personal topics. For years I've been building the "total electronic environment" a piece at a time. When you read some of the excerpts in "**ConnecTime**" perhaps you'll better understand what "under a watchful eye" can really mean.

While my technical background is basically process control and instrumentation, I probably would never have started "computerizing" my home if it were not for the catalyst that started us all thinking about "home control." The invention of the BSR X-10 power line control system ten years ago created a low-cost wireless vehicle for controlling lights and appliances in the home or office.

Originally the X-10 system was introduced as a manual control system whose functions were coordinated through a small push-button controller and a short-range ultrasonic remote. I introduced readers to their first computerized "home-control" project and the X-10 system with a project called "**BusyBox**" (reprinted in Ciarcia's *Circuit Cellar* Volume 1). The popularity of this project and the industrial interest it precipitated gave us an immediate and positive indication that we had picked a significant and timely topic. Thousands of **BusyBoxes** were eventually built.

Since then, X-10 has also come to recognize that automated computer control is a market that they should not ignore. With Ken Davidson's article on using the new OEM X-10 power line interface module, *Circuit Cellar* continues the tradition of being the first place to go for fully supported and documented technology.

As for the "magical man in the box," it's a surrealistic image. It's not that control actions performed by computers should be magic. Instead, we should strive to gain enough knowledge of the technology that we are comfortable in not having to be aware of every event or action they take. Like the "invisible" personal computer which has evolved into an appliance configured for word processing or database retrieval, well-designed home control systems will eventually blend into the environment and merely enhance living. *Circuit Cellar INK* is an application journal dedicated to the effective utilization of computers and in-depth technical applications. The cover picture is merely a theme for an ultimate goal.



READER'S I N K

Letters to the Editor

Dear Steve,

I have been a long-time admirer of your articles. I especially enjoy the way you make the introduction of your articles humorous.

I have just received the March/April 1988 issue of Circuit Cellar INK and I am writing to say that I agree fully with two significant points you make:

1. That a successful publication must have a dedicated core audience -- in this case, technical. (page 1).
2. That technical people are multidimensional (i.e., the "Renaissance Man"). (page 21)

To these, I add my own which is:

3. Technical people seem more willing than most to share their hard-learned knowledge. Many technical people, in fact, are practically fanatical about getting others "energized."

As my contribution to this educational process, I offer at no cost two encryption programs (each of which includes several thousand lines of free source code). All I ask for are blank diskette(s) to put the software on and a self-addressed stamped envelope (SASE).

First, I offer a cryptosystem based on two articles by Professor Lester S. Hill which appeared in American Mathematical Monthly. Hill's articles are included with the software -- reproduced with permission. Hill is the first person to have determined an algebraic way to implement a polygraphic cryptographic algorithm. The two most important features of this implementation are:

1. The security desired can be traded off against the speed of crypton.
2. Keys range in size from 24 bits to 387096 bits. This is over four orders of magnitude variability in key size. This probably represents the largest key supported by any public cryptosystem. Compare this with the 56-bit DES!

This cryptosystem is composed of approximately 2500 lines of "C" language source (written for the

Borland Turbo C compiler).

Second, I also offer a Galois Field Cryptosystem based on an article by Professor Rodney Cooper which appeared in Cryptologia. Cooper's article is included with this software -- reproduced with permission. Cooper enhanced Hill's algorithm in several ways. This implementation also allows the tradeoff between security and speed. Keys range in size from 8 to 9000 bits (this is over three orders of magnitude variability in key size). This cryptosystem is composed of approximately 3000 lines of Pascal language source (written for the Borland Turbo Pascal compiler).

To get your free copy of either (or both) cryptosystems, send one (or two) formatted blank **DS/DD 360kb 5.25-inch** diskette(s) compatible with the IBM PC family of products. Both source code and executable code are contained on the diskettes. Be sure to enclose a self-addressed stamped envelope (SASE) (or, if international mail, please send coupons for stamps) so that the diskette(s) can be returned to you. Mail the diskette(s) and SASE directly to me.

I know, Steve, that you've written about **hardware-based encryption**, but I think you will find that a software-based approach allows for simpler implementation of more robust algorithms by more people. I hope this software encourages education and research, just as this publication does.

Tony Patti
9755 Oatley Lane
Burke, VA

Dear Readers.

While I would ordinarily post this cryptosystem software on the Circuit Cellar BBS for your convenience, there is a federal law which prohibits the export of encryption software or systems without government approval. Since many foreign readers frequent our BBS to download files and communicate with other readers, we might have a legal problem posting such materials. If you are interested in Mr. Patti's offerings, please communicate with him directly.

-- Steve

Dear Steve,

There is a God in Heaven, truly! I knew if I waited long enough someone, somewhere would publish a tech magazine worth reading. I have become hopelessly bored with other publications which seem to be hung up in an endless "for to" loop that contain only benchmarks, Fouriers, and an inexhaustible number of product reviews for '286, '386, "Apple-Mats," "Amiga-Macs," and "Big-Macs"(or is that "a quarter pound of ground round"?).

What a novel idea for Ciarcia & Co.: rocket launchers for soda pop bottles! I actually think that Mr. Nisley is well on his way to solving the national crisis of highway littering. Being a hobbyist of the first degree (that probably makes me a "Hechie" as opposed to a "Techie"; that's "Hechie" not hippie) I plan to enhance this by installing a pointed nose cone with a second stage and launching my discarded bottles into permanent orbit!

All joking aside I found the article both entertaining and educational, along with the rest of the magazine. There is a little child in all of us and you have my sincere thanks for injecting fun back into the high-tech computer arena. What else can I say? Keep the direction; great stuff!

Thomas R. Barnett Jr.
Las Vegas, NV

Dear Steve,

A quick note regarding the Circuit Cellar Neighborhood Strategic Defense Initiative article in the latest issue. The pictures that were published show a shroud covering the upper surface of the airframe (when in launch attitude). What are you hiding? The shroud looks like the sort of cover that the military uses to hide militarily sensitive equipment from prying eyes. Since the airframe sans shroud is a much more aerodynamic form, I can only conclude that you have something to hide.

David Bryant
Arlington, MA

P.S. Could you have been lofting secret payloads by hiding them under the shroud?

Dear Dave,

Cute, Dave. Cute.

Actually, there has been a lot of good response to this article and everyone seemed to appreciate our tongue-in-cheek approach to learning physics. I'm glad we can still have a sense of humor in this high-tech world.

Two questions which many people asked after reading the article were why we didn't use optical sensors instead of wires as gantry sensors and could we publish a more detailed view of the launch mechanism. In answer to the former, believe it or not, typical opto-sensors are too slow for this application. Remember we are blasting past all the sensors in about 2 milliseconds! Regarding the launch mechanism, we'll publish a more detailed picture in an upcoming issue.

-- Steve

Dear Steve,

I am delighted with the content and direction your new magazine is taking. There are too few magazines for the hands-on hardware builder/user these days and now you have given us a new magazine with the flavor and format which others have long ago lost or abandoned. Thanks for giving us "Techies" a long-desired magazine.

David R. Swayer
Edmeston, NY ■

BACK ISSUES ARE AVAILABLE

March/April 1988 Issue

Bottle Rockets: The Physics of Projectiles - Ciarcia/Nisley • The Home Satellite Weather Center - Part 2: NTSC Encoder Alignment and System Overview - Voorhees • Build a 4-Channel Temperature Logging and Data Reduction System - Riley • Digitizing Infrared Signals - Nisley

January/February 1988 Premier Issue

Motion Triggered Video Camera Multiplexor - Ciarcia • Build a Videc Handscanner/Identifier - Nisley • The Home Satellite Weather Center-Part 1: RGBI to NTSC Converter - Voorhees • Keyboard Scanning Subroutines - Nisley

Send \$3.00 plus \$1.00 for postage and handling for each issue you are requesting in check or money order to:

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

Power-Line-Based Computer Control

The X-10 PL513 Power Line Interface Module

by Ken Davidson

We all know how inconvenient it can be to run wires all over the house. The cable TV has to go into the bedroom; you need a telephone in the kitchen, living room, den, and bedroom; signals from the infrared motion detectors have to be sent back to the computer or alarm system... you get the picture. The last thing we need is to run more wires to remotely control lights and appliances in the house. In addition, running 3-conductor #14 wire carrying 110 VAC is a bit more inconvenient and dangerous than running low-voltage twisted pair for an extra phone.

In 1978, BSR introduced a line of modules and control consoles that used the existing AC power lines for the transmission of control signals from the control console to the remote modules. Each module contained either an electronic circuit for turning lights on and off in addition to dimming them, or a mechanical switch for turning appliances on or off. Dubbed "X-10," it soon became a favorite among couch potatoes and consumer electronics junkies alike. It is still private labeled and sold by retail outlets such as Radio Shack (Plug 'n Power), Sears, and others. While the system was great for manually controlling lights and appliances all over the house from a single location, the one feature it lacked was some form of automatic control.

Since its introduction, the X-10 line has gone through marketing changes (it was dropped by BSR and

picked up by X-10 (USA) Inc.) and design changes (the entire line was redesigned several years ago to be more reliable and economical), but until recently still lacked a convenient interface to some form of automated control. True, timers have been part of the product line for quite a while, but they only allow simple on/off commands based on the time of day.



PL513 and remote appliance modules

A "computer interface" was added to the line a few years ago which allows a user with a personal computer to program on/off patterns based on day-of-week in addition to time-of-day. Communication between the computer and the interface takes place over a low-speed serial connection. While a step in the right direction, the programmer still didn't have direct access to the power line. Designers of home control systems who wanted to include X-10 support had to build their own power line interfaces so they could send commands directly to the power line.

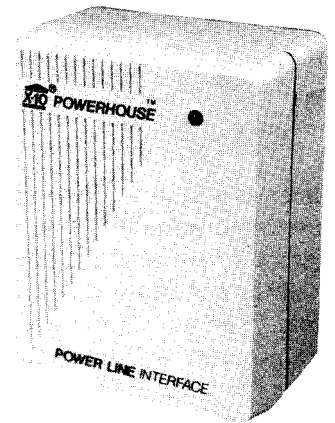
One such control system was the Home Run Control System, designed

and described by Steve Ciarcia in the April-July '85 issues of BYTE. The HCS had the capability to control both X-10 modules and direct outputs based not only on time-of-day and day-of-week, but also on direct inputs from devices such as motion detectors or contact closures. One of the biggest problems faced in that design was how to interface to the power line to send X-10 commands. Isolating the hazardous power line from the comparatively delicate digital circuitry (and delicate human being) is no simple task. A design using a customized wall-mounted transformer, multiple signal wires to the transformer, and coils to inductively couple the transmitted signal to the power line was eventually used.

Such interface headaches are a thing of the past with the introduction of a new power line interface module by X-10. The PL513 power line interface module allows OEMs (and in this case, experimenters) safe access to the power line so their equipment can send X-10 commands directly to the power line.

The X-10 Format

Before getting into the details of the interface module, a description of how the X-10 system works is in order. X-10 is based on a technique known as carrier current communication. The transmitter sends high-frequency bursts out over the power line, which are then received by



remote switch modules. Depending on the pattern of bursts, the module knows whether or not it is being addressed and what function should be carried out.

The AC power line is a harsh environment full of noise, spikes, and transients. The time when the power line is most quiet is when the 60-Hz sine wave that makes up the AC power is crossing 0V, going either from positive to negative or from negative to positive. This is known as the zero crossing. When the zero crossing is detected by the transmitter, a series of 1s and 0s is sent out to the switch modules. A "1" is represented by a 120-kHz burst and a "0" is represented by the lack of a burst (silence). The switch module knows to start listening to the line for a bit sequence when it detects a zero crossing.

Residential AC power is typically single phase (i.e., there is just one sine wave present on the power line). In this setup, sending the bit sequence once at the zero crossing is sufficient in all cases. More complicated AC systems include three phases (i.e., three sine waves separated in phase by 60 degrees). In order to ensure that the transmitted signal will be received by modules connected to any of the three phases, each bit is sent out three times, once at the zero crossing of

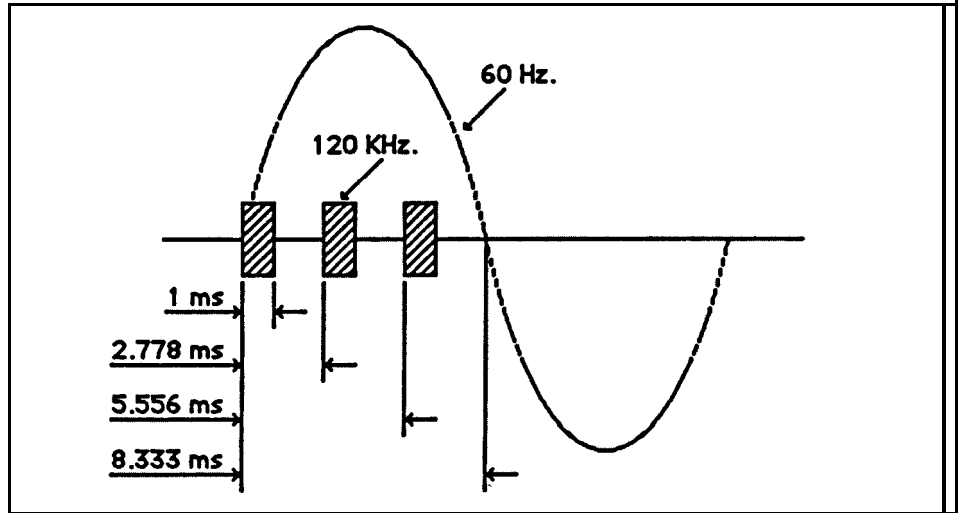


Figure 1 - Timing of X-10 120-kHz bursts for a 3-phase, 60-Hz power line

each of the three phases (figure 1).

The format used by X-10 is an 11-bit sequence consisting of a start code, a house code, and either a module number or a function code (figure 2). To understand why a sequence can only contain either a module number or a function code and not both, you have to understand how a manual X-10 system works. When using the X-10 system from a manual control console, one button is pressed to select a module, followed by a second button to select a function. Hence, two separate sequences are sent.

To ensure that the intended module receives the code sequence without error, it is always sent twice

(except for dim or bright which are sent once).

Whenever a bit is sent to the power line, the complement of the bit is also sent at the next zero crossing. For example, to send a "1" bit to the power line, we wait for a zero crossing, then send out a 120-kHz burst. We then wait for the next zero crossing and don't send anything (denoting a "0"). Another zero crossing must be detected before the next bit can be sent. Using this simple error detecting scheme, the effective bit rate of the X-10 system is 60 bits per second. Needless to say, file transfers between computers over the power lines wouldn't work too well with this system.

The only time the above method isn't used is when sending the start code. The start code must be unique from the rest of the bit sequence so it can be easily detected. It consists of sending "1" bits on three successive zero crossings, followed by silence on the fourth. Figure 3 shows the start code plus the necessary house code and module number bits that would be sent to select module A2.

Once a complete code sequence has been sent, the receiver modules require a silence of at least three power line cycles before the next

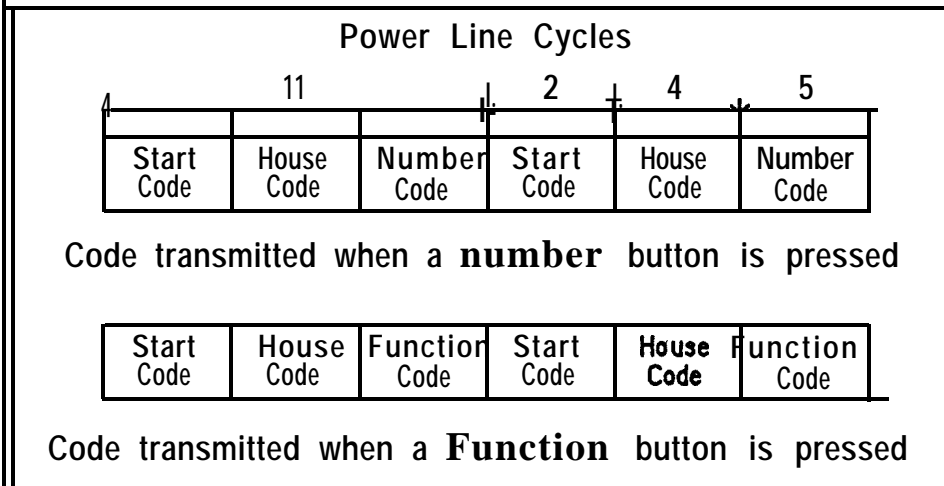


Figure 2 - Format of X-10 sequences

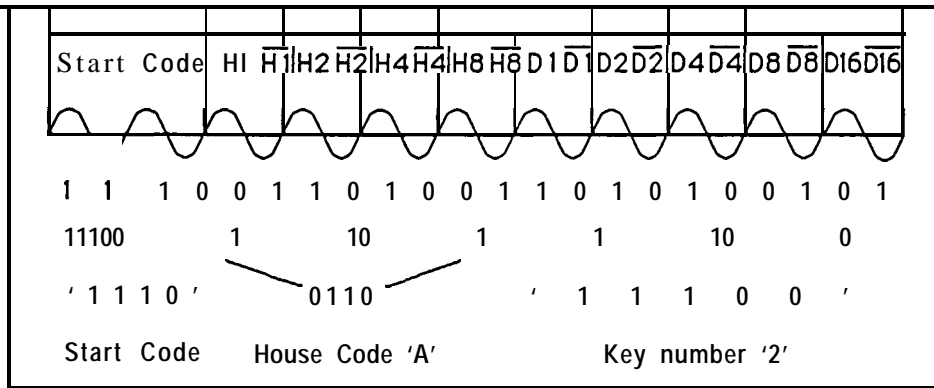


Figure 3 - Bit sequence sent to select module A2

can be sent. "Complete code sequence" here means a complete module select or on/off command sequence. If dim or bright commands are being sent, there shouldn't be a delay until after all the dim or bright commands have been sent, no matter how many there are.

The PL513 Module

Now that we know what the requirements of the code format are, let's take a look at the power line interface module. As you've probably guessed by now, the two functions such a module must provide are a zero crossing detector output to the computer, and an "envelope" input from the computer.

The zero crossing detector puts out a high level when the voltage on the power line is on one side of 0V and a low level when the voltage is on the other side. The computer can look at this signal and wait for the high-to-low or low-to-high transition that signals a zero crossing. When it sees the transition, the computer knows it can start sending its code.

The zero crossing circuit consists mostly of an optoisolator configured with an open-collector output. Since the collector of the phototransistor floats when the transistor is turned off, an external pull-up resistor is required. The emitter of the phototransistor is

connected externally to ground. When the AC neutral line is positive relative to the live line, the LED in the optoisolator is turned on, which causes the phototransistor to turn on. This pulls the collector to ground and the computer receives a low level from the module. When the neutral line is negative relative to the live line, the LED is turned off, which keeps the phototransistor in the isolator turned off, and the collector is externally pulled up to +5V. This generates a high signal to the computer.

The "envelope" input to the module controls whether or not the

signal from the free-running 120-kHz oscillator is sent to the power line. Sending a high level to the module enables the 120 kHz to pass through to the power line, and sending a low level blocks the signal. Such a scheme gives the computer (and programmer) complete control over what is sent over the power line. While it does give the flexibility to send codes other than those defined in the X-10 specification, doing so wouldn't serve much purpose since there isn't anything that could receive them. (However, I'll get into a power line receiver later.)

The oscillator is a simple tuned LC (inductor/capacitor) circuit. The optoisolated envelope signal from the computer gates the oscillator output through a transistor. The final output is coupled to the power line through a coil.

The Computer Connection

Connecting the power line interface module to the computer is the next step. We need one output bit capable of sinking 4.5 mA at +5V, one input bit, a source of +5V

	House Codes				Key Codes						
	H1	H2	H4	H8	D1	D2	D4	D8	D16		
A	0	1	1	0	1	0	1	1	0	0	
B	1	1	1	0	2	1	1	1	0	0	
C	0		01	0	3	0	0		10	0	
D	10		1	0	4	1	0	1	0	0	
E	0	0	0	1	5	0	0		0	1	0
F	10		0	1	6	1	0		0	1	0
G	0		10	1	7	0	1	0		1	0
H	1	1	0	1	8	1	1	0		1	0
I	0	1	1	1	9	0	1	1	1	0	0
J	1	1	1	1	10	1	1	1	1	0	0
K	0		01	1	11	0	0		1	1	0
L	10		1	1	12	1	0		1	1	0
M	0	0	0	0	13	0	0	0	0	0	0
N	10		0	0	14	1	0	0	0	0	0
O	0		10	0	15	0	10		0	0	0
P	1	1	0	0	16	1	10		0	0	0
All Units off					0	0		0		01	
All Lights On					0	0		0	1	1	
On					0	0		10		1	
Off					0	0	1	1		1	
Dim					0	10		0		1	
Bright					0	1	0		1	1	

Figure 4 - House codes and key codes used in the X-10 protocol

for a pull-up resistor, and ground. Connection to the host computer depends entirely upon the I/O resources available on the computer, so I've chosen three specific computers to interface to the PL513: the IBM PC, and the Micromint BCC52 and BCC180. I chose the former because of its popularity and the latter two because they are ideally suited to dedicated power line control through the PL513 module. (The BCC52 was presented in the August '85 issue of BYTE and the BCC180 appeared in the January-March '88 issues of BYTE.)

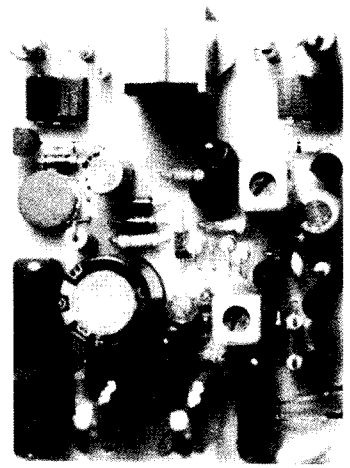
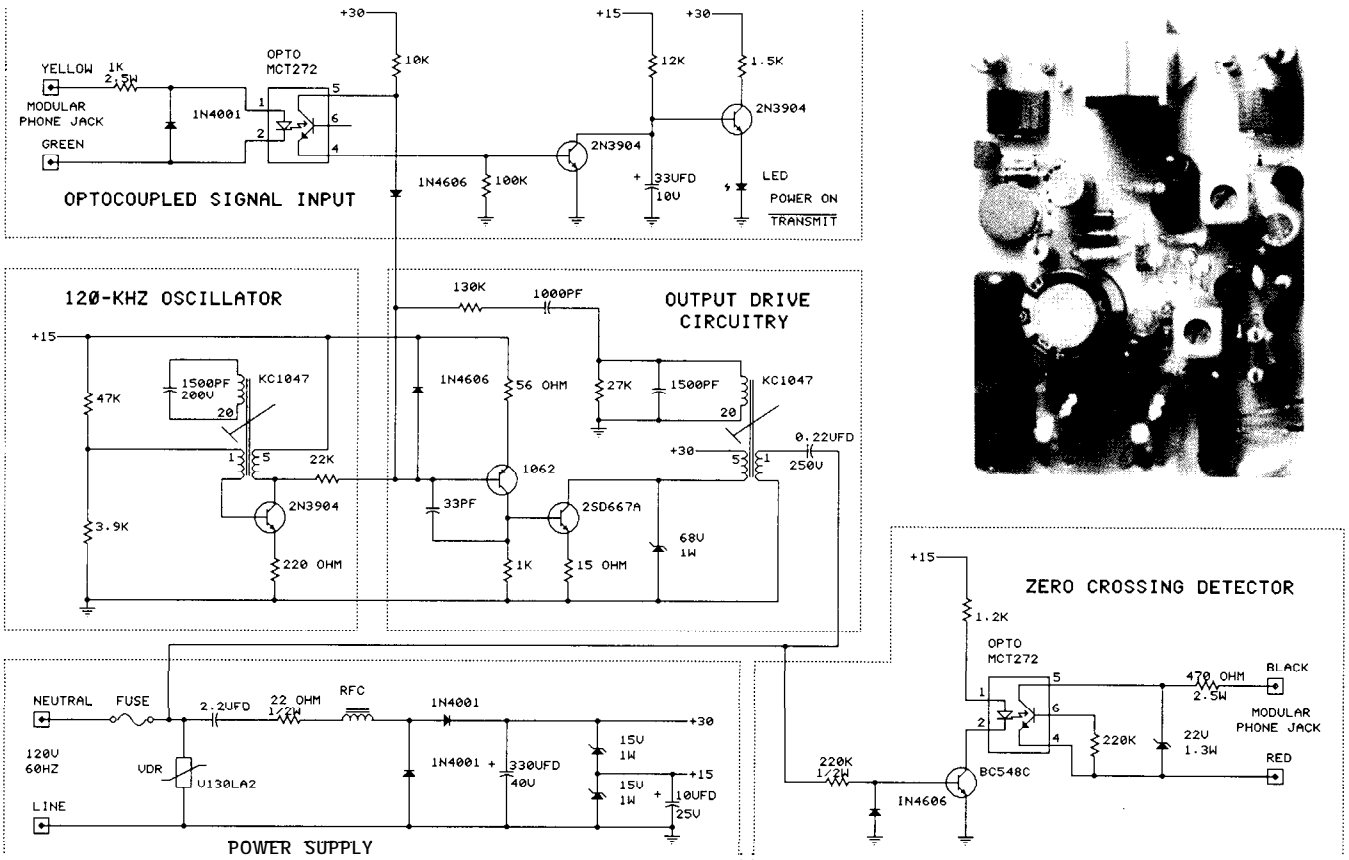
The most convenient interface on the IBM PC for this application is the parallel printer port. It has plenty of output lines plus several input lines. Upon further inspection of the printer adapter's schematics, I found that the -SLCT IN

input is already pulled up to +5V, making it ideal for use as the zero crossing input. Since none of the output lines can sink 4.5 mA alone, I tied four of them together to gain the necessary current capability. Figure 6 shows the adapter necessary to go from the printer port's DB-25 connector to the PL513's RJ-11 jack.

There is one precaution that must be observed whenever using a setup such as this. Four of the driver outputs are tied together. If all four bits are set to the same level, then everything is OK. If any one bit is set differently from the others, two or more drivers will be fighting each other, possibly damaging the printer adapter. Before connecting the adapter shown in figure 6 to the computer, be sure to run the software described later so that the outputs can be properly set up.

The BCC52 and BCC180 interfaces are a bit more involved, but not much. Both interfaces are the same since we are connecting to an 8255 PPI (programmable peripheral interface) present on both boards. Each board has a 26-pin connector dedicated to the three ports on the 8255. Bit 7 on port A is connected to the PL513's zero crossing output. Since the 26-pin connector doesn't have a source of +5V for the pull-up resistor, we'll set up port C to be an output port and leave all its output bits set high. When a resistor is connected between the zero crossing signal and the high bits on port C, we have effectively pulled up the signal.

In connecting the 8255 to the PL513's envelope input, the same issue as with the IBM PC connection arises: none of the output bits on the



Schematic for the PL513 and (photo insert) inside of the module

8255 has enough current sinking capability alone to do the job. So, again, four bits tied together are used to provide the envelope signal. Since all three of the 8255's ports are set for input when the chip is reset, we don't have to worry about two bits fighting each other with conflicting levels after reset as we did on the IBM PC. The X-10 control software will have complete control over the levels set on the tied-together outputs.

Figure 7 shows how an adapter similar to that for the IBM PC can be made to go from the RJ-11 on the module to a DB-25 connector. Figure 8 shows the necessary cable to go from the DB-25 to the 2x13 Berg-type header on the BCC52 and BCC180. While a direct connection can be made between the RJ-11 and the 2x10 header, the adapter/cable pair allows for a very clean, modular connection.

Software

The final step in getting this mess to work is the software. While Steve admits to avoiding software as much as possible, I enjoy writing low-level support code as much as I do designing the hardware. I wrote some code for the BCC180 interface and coerced some colleagues to write the same code for the BCC52 and IBM PC interfaces. Excerpts from that code are shown in Listings 1, 2, and 3.

Listing 1 shows the BCC180 code necessary to send out a bit sequence to the PL5 13 module. The *send* routine is called with register A containing the sequence of bits to be sent and register B containing the number of bits to be sent. Given a house code, module number, or function, I can look up the proper bit sequence in a table and call *send* with that bit sequence. *send* first waits for a zero crossing, then sends out the low-order bit, complements the low-order bit, waits for another

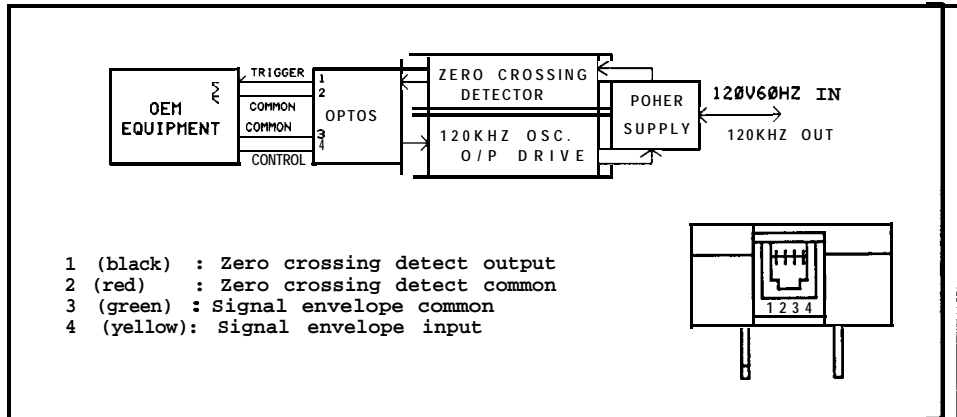


Figure 5 - PL513 block diagram and pinout

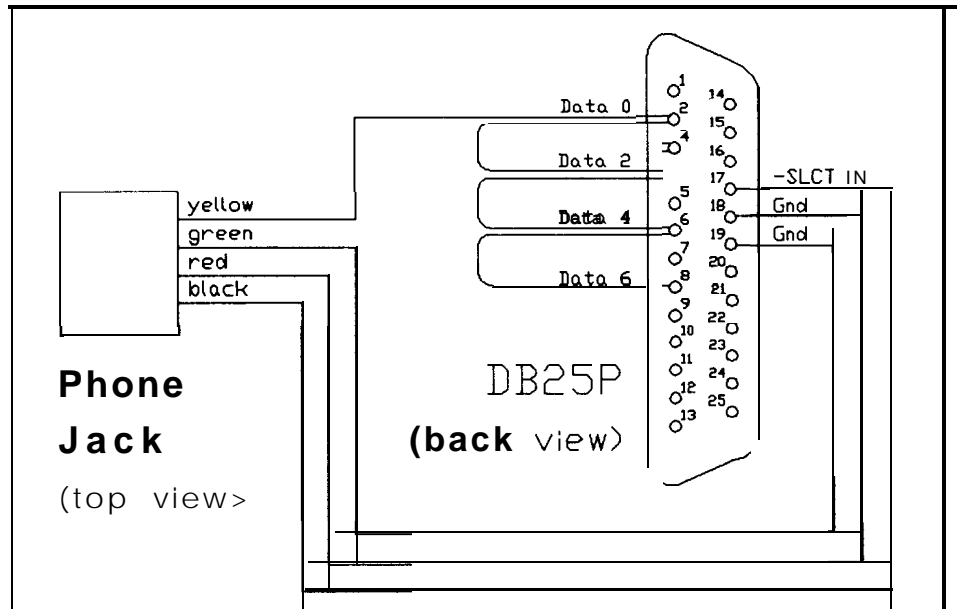


Figure 6 - RJ11-to-DB25 adapter for the IBM PC

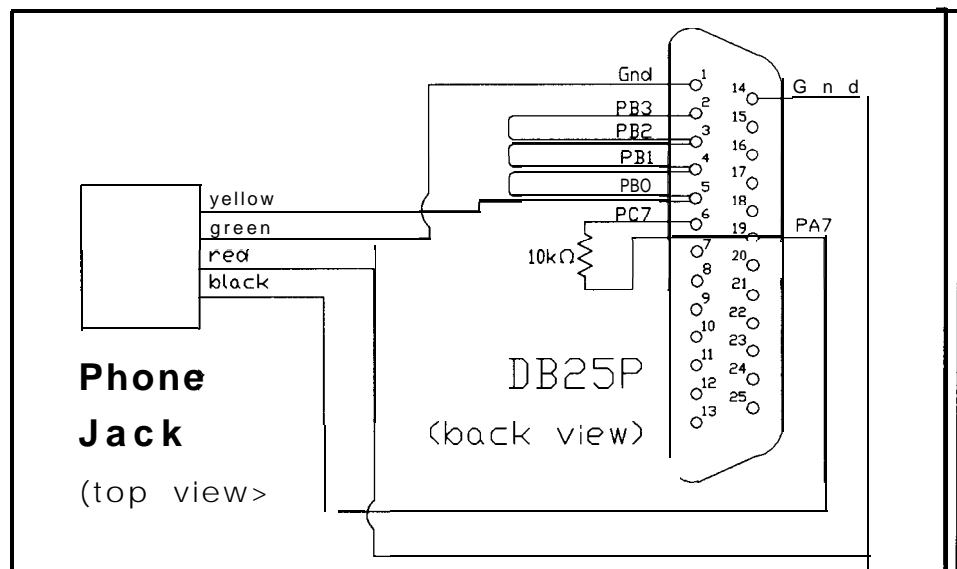


Figure 7 - RJ11-to-DB25 adapter for the BCC180/BCC52

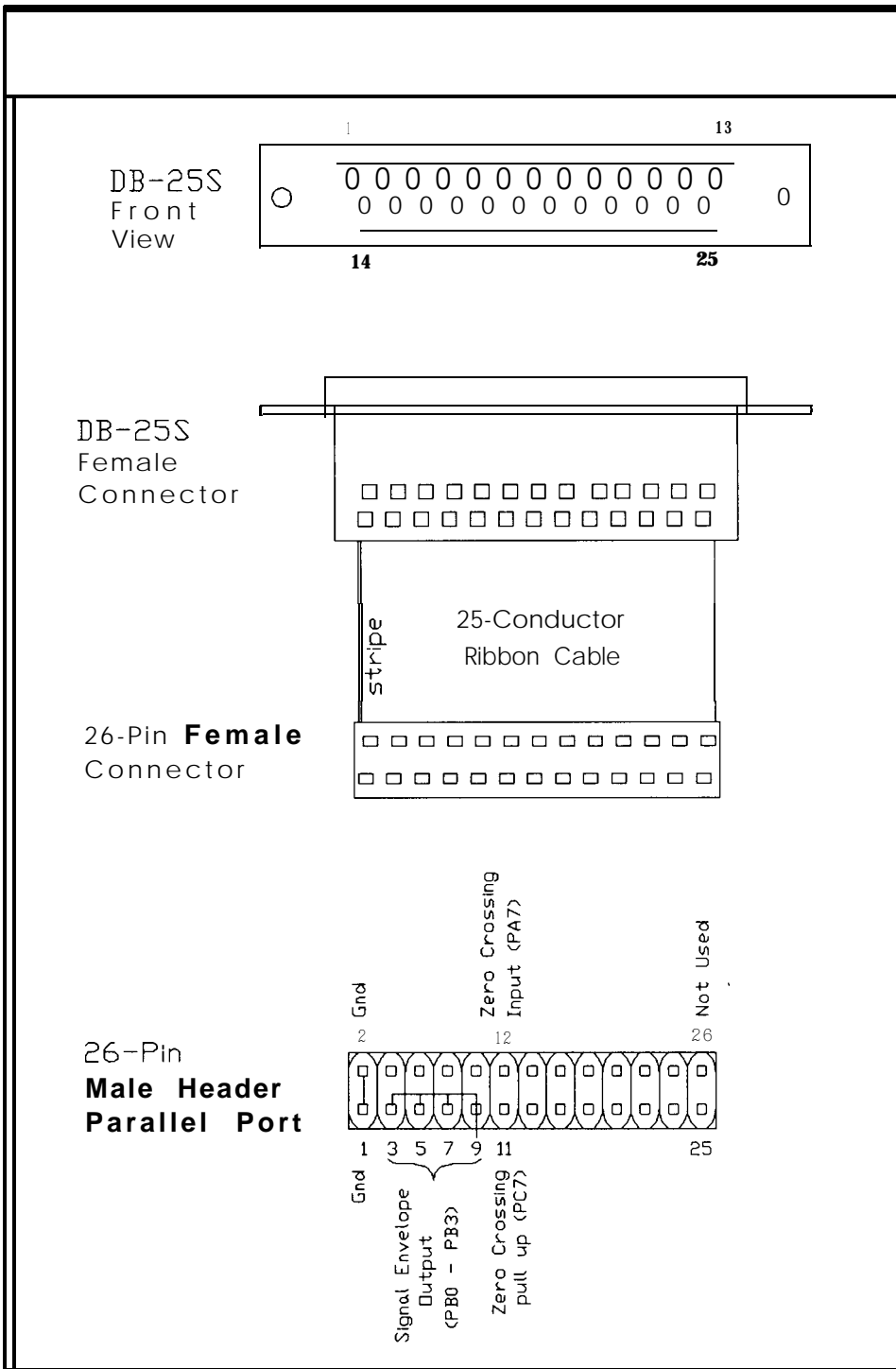


Figure 8 - X-10 / BCC Cable to connect the adapter in Fig. 7 to the BCC180/BCC52 parallel port

zero crossing, sends the complemented low-order bit, complements the bit again to restore it to the original value, rotates the next higher bit to the low-order position, checks to see if it's done, and, if not, does it all again for the next bit.

wtcross is used to wait for the next zero crossing transition to occur. It first reads the current

value and stores it away. It then begins reading new values and comparing them to the original. When a change is sensed (the XOR function yields a nonzero result), a transition has occurred and the routine returns to the caller. A delay is used during the scanning to add some hysteresis and avoid possible noise-induced chattering as the AC signal gets close

to 0V.

The *sendbit* routine is used to send the low-order bit of register A to the power line. It assumes the zero crossing has already been detected. *sendbit* sends out either a 1-ms 120-kHz burst to denote a "1" or 1 ms of silence to denote a "0." It then inserts a delay of 1.778 ms, followed by the same burst or silence. It repeats this once more and returns. Recall that each bit is sent out three times to account for the zero crossings of a three-phase system. The time between the zero crossing of one phase and that of another is 2.778 ms.

Those familiar with Z80 programming won't recognize the "inb" and "outb" instructions. Those are macros used to access a full 16-bit port address. For example, "inb reg,port" is replaced by the instructions "ld bc,port" and "in reg,(c)". The standard "in" and "out" instructions can only be used to access 8-bit port addresses.

BCC52 Software

Listing 2 shows the same routines as described above written for the BCC52. Since a different programmer wrote the code, a slightly different approach was taken toward achieving the same goal. It can be instructive not only to study how the same task is done on different processors, but also how different programmers solve the same problem. *ALTLP* is approximately equivalent to the BCC180's *send* routine, but not quite as modular. *ALTLP* waits for a zero crossing, then rotates the low-order bit into the carry bit and checks whether to send a "1" or a "0." If a "1" is to be sent, a call to *ONEBIT* is made, then it waits for a zero crossing. By doing nothing after the zero crossing, the routine implicitly sends a "0," the complement of the "1" just sent. If a "0" is to be sent, nothing is done but wait for the next zero crossing,

then a "1" is sent. After the I/O or O/I pair is sent, the routine checks to see if it's done and loops if not.

WAITEDGE used to wait for the next zero crossing and is relatively straightforward. It reads the current value of the zero crossing detector and waits for the complement.

ONEBIT is similar to the **BCC180's sendbit**. It sends out three 1-ms 120-kHz bursts spaced 2.778 ms apart.

IBM PC Software

Finally, Listing 3 shows the equivalent code for the IBM PC. As before, since a different programmer did the coding, a slightly different approach was taken to achieve the final goal.

Instead of making a **general-purpose** routine for sending n bits of X-10 code, this programmer wrote the **x10msg** routine to send the entire 11-bit command sequence including start code, house code, and either module number or function code. The house code and module number/function code are passed to the routine in the AX register. Once the start code has been sent, nine more bits plus their complements are sent, each sent by calling **send3ph**.

send3ph (send three phases) is used to send the bit in **bitval** to the power line. When the routine is entered, one of the IBM's timers is cleared immediately after a zero crossing. The timer is checked at different times during the routine to determine proper timing of the 120-kHz bursts and intervals between them.

This programmer has a penchant for macros, and several are used in **send3ph**. **chksync** is shown in the listing and is used to get the current zero crossing value and compare it to the previous value. The zero flag is cleared when a transition is sensed. **tstart** simply starts the timer

Listing 1 - BCC180 code to send a series of bits to the power line

```

;
; Send the byte in A to the transmitter module aerially,
; LSB first. Number of bits to send is in B.
;
send:
    call    wtcross        ; Wait for next zero crossing
    call    sendbit       ; Send low bit
    cpl
    call    wtcross        ; Wait for next crossing
    call    sendbit       ; Send complement of bit
    cpl
    rrca                    ; Rotate next bit into position
    djnz   send           ; Do next bit
    ret

;
; Wait for the next zero crossing.
;
wtcross:
    push   af
    push   bc
    push   de
    inb   a,zxing         ; Get current zero crossing
    and   80h             ; Mask bit
    ld    d,a             ; Save it

wtc1:
    ld    b,20            ; Pause 20 us
wtc2:
    djnz  wtc2
    inb   a,zxing         ; Get new state
    and   80h             ; Mask bit
    xor   d               ; Has it changed?
    jr    z,wtc1         ; No, keep waiting
    pop   bc
    pop   af
    ret

;
; Send bit 0 of A to the transmitter. Sends it three times,
; once for each power phase.
;
sendbit:
    push   bc
    push   de
    ld    e,0             ; Assume LSB is 0 for now
    bit   0,a             ; Check LSB
    jr    z,sndb         ; Zero, OK
    ld    e,0ffh         ; One, turn on all bits

sndb:
    ld    d,0
    outb dout,e          ; Send out bit
    call dlylms          ; Wait 1 ms
    outb dout,d          ; Turn bit off
    call dlyphas        ; Wait 1.778 ms
    outb dout,e          ; Send out bit
    call dlylms          ; Wait 1 ms
    outb dout,d          ; Turn bit off
    call dlyphas        ; Wait 1.778 ms
    outb dout,e          ; Send out bit
    call dlylms          ; Wait 1 ms
    outb dout,d          ; Turn bit off
    pop   de
    pop   bc
    ret

```

after a zero crossing has been **detected**. **waitfor** will constantly check the timer until the time specified as a parameter is reached.

Just the beginning

This is hardly the end of our use

of the PL513 power line interface module. Steve and I have already started discussing a possible bus-oriented "HCS II" project based on a BCC180 and the PL513. Be assured that as details of this project unfold, you will be kept apprised.

Also, as a result of our interest

Listing 2 - BCCS2 code

```

;
;*****
;
; ALTLP - SEND ALTERNATING BIT PATTERNS TO 8255 PORT
;
;*****
;
ALTLP EQU $
      CALL WAITEDGE      ; WAIT FOR NEXT EDGE
      RLC A              ; GET NEXT BIT TO SEND INTO CARRY
      JNC ZEROONE       ; IF ZERO, SEND ZERO THEN ONE
;
; SEND ONE THEN ZERO
;
ONEZERO EQU $
      CALL ONEBIT        ; SEND ONE FIRST
      CALL WAITEDGE     ; WAIT FOR NEXT EDGE
                        ; SEND ZERO BY DOING NOTHING
      JMP ALTEND        ; CHECK FOR ALL DONE
;
; SEND ZERO THEN ONE
;
ZEROONE EQU $
      CALL WAITEDGE     ; SEND ZERO BY DOING NOTHING
      CALL ONEBIT       ; SEND A ONE
;
ALTEND EQU $
      DJNZ B,ALTLP      ; IF MORE BITS DO THEM
      RET              ; ELSE RETURN TO CALLER
;
;*****
;
; ONEBIT - SEND ONE-BIT PATTERN TO 8255.
;          SEND THREE 1.0-MS BURSTS SEPARATED BY 1.778 MS
;
;*****
;
ONEBIT EQU $
      PUSH ACC          ; SAVE A REG CONTENTS
      PUSH B            ; AND B REG TOO
      MOV B,#3          ; DO PULSE 3 TIMES
      MOV DPTR,#X10OUT ; POINT TO OUTPUT PORT
;
; SEND 1-MS PULSE AND WAIT 1.778 SECONDS
;
PULSELP EQU $
      PUSH B            ; SAVE B COUNT
      MOV A,#0FH       ; LOAD A WITH ONE'S PATTERN
      MOVX @DPTR,A     ; SEND TO FORT
      MOV B,#LOOP1000 ; GET DELAY VALUE
      CALL DELYLP      ; DELAY 1 MS
;
      MOV A,#00H       ; GET ZERO BIT PATTERN
      MOVX @DPTR,A     ; SEND TO FORT
      MOV B,#LOOP1778 ; GET DELAY VALUE
      CALL DELYLP
      FOP B            ; GET COUNT
      DJNZ B,PULSELP  ; DO ALL 3 PULSES
;
      POP B            ; RECOVER SAVED
      POP ACC          ; REGISTERS
      RET              ; A N D R E T U R N T O C A L L E R
;
;*****
;
; WAITEDGE - WAIT FOR THE NEXT TRANSITION OF THE ZERO
;            CROSSING SQUARE WAVE FROM THE INTERFACE
;            MODULE.
;
;*****
;
; FIND OUT WHAT LEVEL'S HERE NOW, AND WAIT FOR THE OTHER ONE

```

(continued on page 12)

in the PL513 and a certain amount of prodding, X-10 is developing a two-way PL513 module capable of not only transmitting X-10 codes onto the power line but also receiving them. What was once a strictly open-loop system that was essentially blind can now become closer to a closed-loop system. While the computer won't be able to poll each module for its current state, it can keep track of X-10 activity taking place on the power line initiated by sources other than itself. If the computer automatically turns the light in the living room off, but I'm sitting there reading (now in the dark) and manually turn the light back on, the computer will be able to "hear" the command as I send it to the lamp module and will know not to turn it off again.

Another potential use for such a receiver is to use standard X-10 control consoles as very low-cost, limited-function remote terminals to the computer. I can set a controller to a different house code than that used by the modules around the house. I can then define the "1" button to mean, "I'm leaving the house. Turn off all the lights and arm the alarm." I can define button "2" to mean, "I'm going to bed. Disable the motion detectors in the bedroom." Or I can define the "4" button to mean, "Dim the four lights in the living room to some predefined levels." The potential use of such a receiver in home control is limitless. We will be sure to keep you up to date as to its status as development unfolds. ■

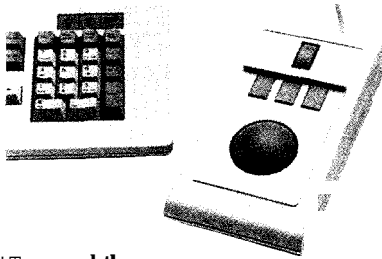
Special thanks to Dave Rye, Bill Curlew, and Ed Nisley for their contributions to this project.

Diagrams and schematics pertaining to the PL513 are reprinted by permission of X-10 (USA) Inc.

PL513 modules may be obtained directly from X-IO (USA) Inc. Software described in this article may be downloaded from the Circuit Cellar BBS.

FastTrap™

The Pointing device of the future is here!

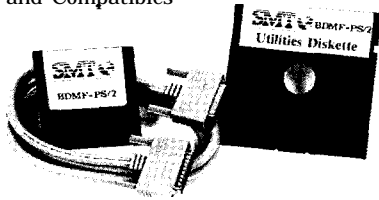


- *Two and three axis pointing capability.
- *High resolution trackball for X and Y axis input.
- *High resolution fingerwheel for Z axis input.
- *Use with IBM® PC's, XT's, AT's and compatibles.
- *Three input buttons.
- *Full hardware emulation of Microsoft® Mouse.
- *Standard RS-232 serial interface.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

ONLY \$149.00

THE INTERCHANGE™

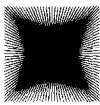
Bi-directional Data Migration Facility for IBM PS/2, AT, PC, PORTABLE and Compatibles



Featuw:

- *Parallel port to parallel port.
- *Economic method of file transfer.
- *Bi-Directional tile transfer easily achieved.
- *Supports all PS/2 systems (Models 30, SO, 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



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

Listing 2 (continued from page 11)

```

;
WAITEDGE EQU $
          PUSH ACC           ; SAVE A JUST IN CASE
          PUSH DPH          ; AND DPH
          PUSH DPL          ; AND DPL TOO.
          MOV DPTR,#X10IN   ; POINT TO INPUT PORT
          MOVX A,@DPTR      ; GET CURRENT LEVEL
          ANL A,#80H        ; ISOLATE ZERO CROSSING BIT
          JZ WAITONE        ; IF ZERO, WAIT FOR ONE

;
; WAIT FOR ZERO LEVEL
;
WAITZERO EQU $
          MOVX A,@DPTR      ; GET CURRENT LEVEL
          ANL A,#80H        ; ISOLATE ZERO CROSSING
          JNZ WAITZERO      ; IF ONE, WAIT FOR ZERO
          JMP EDGEEND       ; IF ZERO, ALL DONE

;
; WAIT FOR ONE LEVEL
;
WAITONE EQU $
          MOVX A,@DPTR      ; GET CURRENT LEVEL
          ANL A,#80H        ; ISOLATE ZERO CROSSING
          JZ WAITONE        ; IF ZERO, WAIT FOR ONE

;
EDGEEND EQU $
          POP DPL           ; RESTORE THE
          POP DPH          ; SAVED
          POP ACC          ; REGISTERS
          RET              ; ANDRETURW TO CALLER

```

Listing 3 - IBM PC code

```

;-----
; Test sync input for an edge
; Assumes DX is set up with control port address

chksync MACRO
          IN AL,DX           ; get current sync bit
          XCHG ctlmem,AL    ; save for next time
          XOR AL,ctlmem     ; different. from old value?
          AND AL,MASK syncbit ; isolate correct bit
          ENDM

;-----
; Sends a complete X10 message from AX
; Waits for next zero crossing, whenever that may occur
; Uses bitval and bitstream to stash successive bits

x10msg PROC NEAR

          MOV bitstream,AX  ; set up local copy of bits
          PUSH CX           ; save bystander

;--- send the start code (1110)

          MOV bitval,OFFH
          CALL send3ph
          CALL send3ph
          CALL send3ph
          MOV bitval,0
          CALL send3ph

;--- mail off the message bits

          MOV CX,(WIDTH house) + (WIDTH key)
nextbit LABEL NEAR
          SHL bitstream,1   ; set up bit to send
          MOV bitval,0      ; assume 0 bit
          TEST bitstream,MASK headbit
          JZ sendpair
          NOT bitval        ; oops... make it FF

```

(continued on page 13)

isting 3 (continued from page 12)

```

endpair LABEL NEAR
CALL send3ph ; send bit true...
NOT bitval ; and complement
CALL send3ph
LOOP nextbit ; repeat for all bits
POP CX ; restore bystander
RET
10msg ENDP
    
```

 Send three copies of bit at three-phase zero crossings
 Waits for next zero crossing and restarts timer
 Gets bit value from global bitval
 Mashes BL and DX

```

end3ph PROC NEAR
--- reset timer, wait for zero crossing, start timer

CALL tinit
MOV BL,bitval ; set up bit value for loop
MOV DX,ctlport ; set up current sync input
IN AL,DX
MOV ctlmem,AL
    
```

```

eget LABEL NEAR
chksync
JZ reget ; loop if no edge
tstart ; crossing! start timer
    
```

--- tell them three times...

```

MOV DX,dataport
MOV AL,BL ; phase A
OUT DX,AL ; ... bit active
waitfor Time1000us
MOV AL,0 ; ... and inactive
OUT DX,AL
    
```

waitfor Time2778us

```

MOV AL,BL ; phase B
OUT DX,AL
waitfor Time2778us+Time1000us
MOV AL,0
OUT DX,AL
    
```

waitfor Time5556us

```

MOV AL,BL ; phase C
OUT DX,AL
waitfor Time5556us+Time1000us
MOV AL,0
OUT DX,AL
    
```

RET

end3ph

WRITE FOR INK!

finish t h a100-MIPS
 Or,

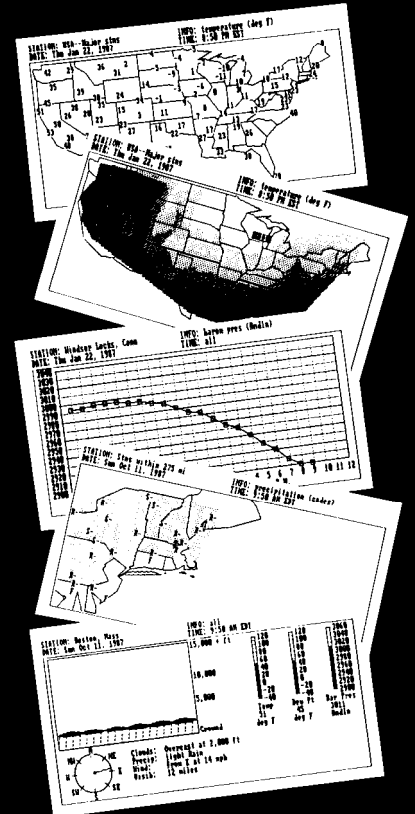
Harv y ~~Miniprojectontline~~ to c u i t Cellar INK, PO Box 772,
 Vernon, CT 06066, or contact him on the Circuit Cellar BBS at (203) 871-1988.

POWER!

The Accu-Weather Forecaster is a powerful new computer program that enables you to receive **live weather information** direct from Accu-Weather, the largest private weather service in the country. This same information is used by professional meteorologists in preparing their own forecasts. And now, you can possess it too!

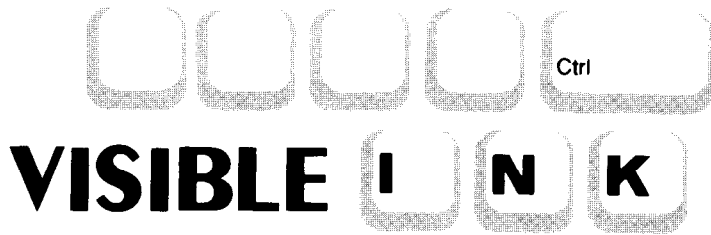
- Fully automated telecommunications
- Menu-driven format
- Only \$2.00 per typical download
- Informative weather screens, including maps, graphs, pictures and forecasts
- Official U.S. Summary prepared four times a day
- Includes program disk, sample data disk, User's Manual, Forecasting Guidebook

Yes, please send me _____ copies of
The Accu-Weather Forecaster
 at **\$89.95**
 plus \$2.00 postage and handling per order.
 Conn. state residents add 7.5% sales tax.
 Macintosh and IBM formats (**Modem required**)
 MasterCard, Visa or check acceptable
 For more information, call (203) 223-5911



Accu-Weather FORECASTER

From Metacom Software, Inc.
 P. O. Box 31337 Hartford, Connecticut 06103
 Accu-Weather is a registered trademark and Accu-Weather Forecaster is a trademark of Accu-Weather, Inc., which was cooperatively involved in the development of this program.



Answers; Clear and Simple

Letters to the INK Research Staff

Dear INK,

In thumbing through the back pages of magazines, I note that the cost of an 8088 chip is about \$8.00, the cost of an 80286 chip is between \$15.00 and \$35.00 (depending on speed), and the cost of an 80386 chip is about \$450.00. What explains this vast difference in price? Greater precision? Lower production quantity? Greater demand and fewer substitutes?

Howard E. Abrams-Clarkston, GA

Dear Howard,

It is true that the 80386 is much more powerful in terms of architecture, speed, and capability than earlier chips such as the 8088, but the cost differential is not entirely due to this difference. The Intel technology family of microcomputer chips has evolved from the 1008 in the early 1970s to the present 80386 processor in 1988. If we look at the cost of an 8008 chip back in '971-72 we would see they were selling for approximately \$200.00 in small quantities.

Accounting for inflation, this equates roughly to the cost of the 80386 today. The chip developers try to recoup their research and development costs during the first few years of production, with prices lowering as the chip matures and begins to see volume production. The 80386 chip may never see a price of \$15.00, but I think you will see large reductions from present prices as the chip begins to experience volume production.

-- INK

Dear INK,

I've decided to try a little hacking of my own. The result is the creation of a video display board for the Apple and Commodore group of computers. I'd like to publish the circuit and produce a kit, but I'm afraid my board will run afoul of the FCC RFI Standards (specifically Part 15) established for computer peripheral devices. Any comments?

Greg Clark-Sacramento, CA

Dear Greg,

You should remember that the FCC is a bureaucracy that, like all bureaucracies, exercises its power somewhat arbitrarily and inconsistently. Any electronic device that causes sufficient interference with radio and television reception to cause complaints by neighbors, for instance, is likely to run a foul of the FCC, regardless of anything else that may apply in the way of permits, licenses, and regulations.

Officially, the only products requiring FCC Part 15 certification are factory-assembled units in cases: especially those deemed as "personal computer related."

Partially assembled units, boards, and kits fall into a grey area as far as the regulations go. The FCC mentioned to us that Heathkit and Circuit Cellar Inc. are among the very few manufacturers who certify kits. However, the FCC doesn't appear to be pursuing the kit makers and magazines in any deliberate or systematic way. As a practical matter, then, kits and OEM boards do not seem to need FCC certification, but if you market a design that has interference problems the FCC may get interested.

-- INK

Dear INK,

Can you recommend a place that I can have PC boards made from smARTWORK plots? I currently pay \$0.63 per square inch for PCBs but they don't do through-hole plating. (The only place I found that does, wants \$200 for boards I presently have made for \$7).

David Johnson-S, Hempstead, NY

Dear David,

The vast price differences in printed circuit services, which you note in your letter, make us all wonder what criteria is used for pricing, and whether using a cheaper service will yield poor results.

One of our staff members knows of a service in North Carolina, which not only will do the circuit work from smARTWORK plots, but is able to take your

smARTWORK file via modem and create boards at a very reasonable price (recently they quoted double-sided, plated-through-hole boards made in small quantities for less than \$1 / sq. in., plus setup). For more information, contact: Express Circuits, P.O. Box 58, Wilkesboro, NC, 28697, (919) 667-2100.

-- INK

Dear INK,

I have searched high and low trying to find out how to perform a low-level format on an IBM PC (not AT) clone's hard drive, but no luck. It's not in the manuals and it's not in any of Peter Norton's books that I have read. I know it can be done with DEBUG, and you can specify whatever interleave factor you want.

While I'm at it, is a hard drive controller's ability to operate at a given interleave factor affected if the CPU's clock speed is changed? Must a controller have Full-track buffering in order to provide 1:1 interleave?

Richard F. White-Washington, DC

Dear Richard,

The reason you can't find the low-level formatting information anywhere is because it's different for each and every hard drive controller! Honest!

For example, a genuine True Blue IBM PC/XT doesn't have the formatting routine in ROM -- you've got to dig out the diagnostics diskette and start from there. In fact, you might have had to shell out for the Advanced Diagnostics disk to get the formatter.

Most clone manufacturers found that there was plenty of room left in the ROMs on the controller board to hold a simple formatting routine. Unfortunately, they didn't get together beforehand, so there's no standardization on how to start the routines, how to tell them what to do, or how they return control. They run the gamut from full-screen displays to cryptic DEBUG interfaces with hex values in registers.

If you didn't get any information with your controller, you're out of luck. There's no (practical) way to snoop around in the ROM to figure out how to start it up.

And, yes, the interleave depends on the clock speed. The faster your processor runs the sooner it's ready for the next sector, so the interleave can be smaller. If it's fast enough, you can get 1:1 interleave without a track buffer: not on a PC, but surely on a hyperthyroid AT.

The penalty for making the interleave too small is that the sector is ready before the processor can accept it. No harm done, but by the time the CPU asks for the

sector, it has to wait for nearly a whole revolution. If the interleave is too large, it has to wait only a fraction of a revolution: as with airplanes, it's better to be a whole lot early than a little bit late

-- INK

Dear INK,

I am very interested in artificial intelligence and I have a few questions.

First, I would be very thankful if you could point out the differences between the LISP and Prolog languages, mainly focusing on their different uses. Could you also recommend an introductory book on each language as well as introductory compilers (or interpreters) for each language?

Also, could you recommend books on the following topics: database design (including concepts of relational databases; introductory guide to the language C; introduction to the concepts and techniques involved in programming Heuristic Algorithms; and a good guide to the concepts of artificial intelligence (not to introductory since I already have an introductory text).

Finally, could you recommend a good C compiler to start with (maybe Turbo-C by Borland)?

Anthony J.D. Wheeler-France

Dear Anthony,

The field of Artificial Intelligence is growing by leaps and bounds today. It would be very difficult for anyone who is seriously interested in Computer Science to ignore the subject.

Today's AI languages such as LISP, Prolog, and OPS5 are primarily being used to develop what are known as "Expert Systems." These software systems embody the knowledge of a person, or group of people referred to as the Domain Expert, in a computer program which is usually used by someone much less experienced in the particular domain of knowledge used. The expert system has in a sense been "taught" many of the things the human expert knows.

Typically, expert systems are based on heuristics as opposed to straight-forward algorithmic programming solutions. Heuristics are essentially "rules of thumb" which we know or believe to be true. These rules of thumb are coded into the program and become a part of what is referred to as the knowledge base. An expert system is made up of rules or productions. A rule is often an IF-THEN type of statement that tests the data being examined. For example the following rule coded in

OP55 performs a test to determine if an animal under test is a carnivore:

```
(literalize animal
      teeth
      eyes
      type
)
( p Is-Carnivore
  ((animal ^teeth pointed ^eyes forward ^type null)
  <animal>)
  -->
  (modify <animal> ^type carnivore)
)
```

The beauty of AI languages is that their syntax has been designed to facilitate analyzing nonnumerical, nontext data. In most cases one of the AI languages may be used to develop a model of a problem whose solution relies on heuristics much faster than a traditional computer language like BASIC, FORTRAN, or C. Several books which may be of interest include:

- *The C Programming Language* by Brian Kernighan and Dennis Richie ISBN 0-13-110163-3
- *MIX Software* 2116 E. Arapaho Suite 363 Richardson, TX 75801 (214) 783-6001
- *Building Expert Systems* Edited by Hayes-Roth, Waterman, Lenat, 1983 Published by Addison Wesley
- *Heuristics* by Pearl Published by Addison Wesley

As for compilers or interpreters for various AI languages, it rather depends on what type of computer you have access to. Versions of most major languages are available for MS-DOS-based computers but we are unable to give you a recommendation on any particular one.

Information on database concepts and techniques should be available in nearly any school library. MIX Software offers a complete C compiler and linker for either MS-DOS or CP/M computers for under \$20.00. The system comes with a decent tutorial as well and is a nice system for learning C overall.

-- INK

Dear INK,

I like to build interface cards for my IBM XT clone computer.

One thing that I would really like to learn as much as possible about is the Hewlett Packard interface bus HPIB (IEEE-488). We use HP structural analyzers at work and I have written programs on HP9000 computers (in HP BASIC) to act as a controller for the analyzers.

I would like to make an HPIB bus but they describe the bus through a user's standpoint. What I'm looking for is something that describes the HPIB from the component level with good timing diagrams and maybe even example schematics. The HP representative sent me a manual "Tutorial Description of the HPIB" but it's mostly about programming on the bus.

If you could direct me to some good information on the subject I would appreciate it.

Rick Smith-Lansing, MI

Dear Rick,

As a hardware builder/enthusiast, you may be one of a neglected class. While the earlier days of microcomputing were characterized by users who knew something about their hardware (and cared,) the present sometimes seems increasingly dominated by appliance operators who can't tell the difference between a computer and a toaster.

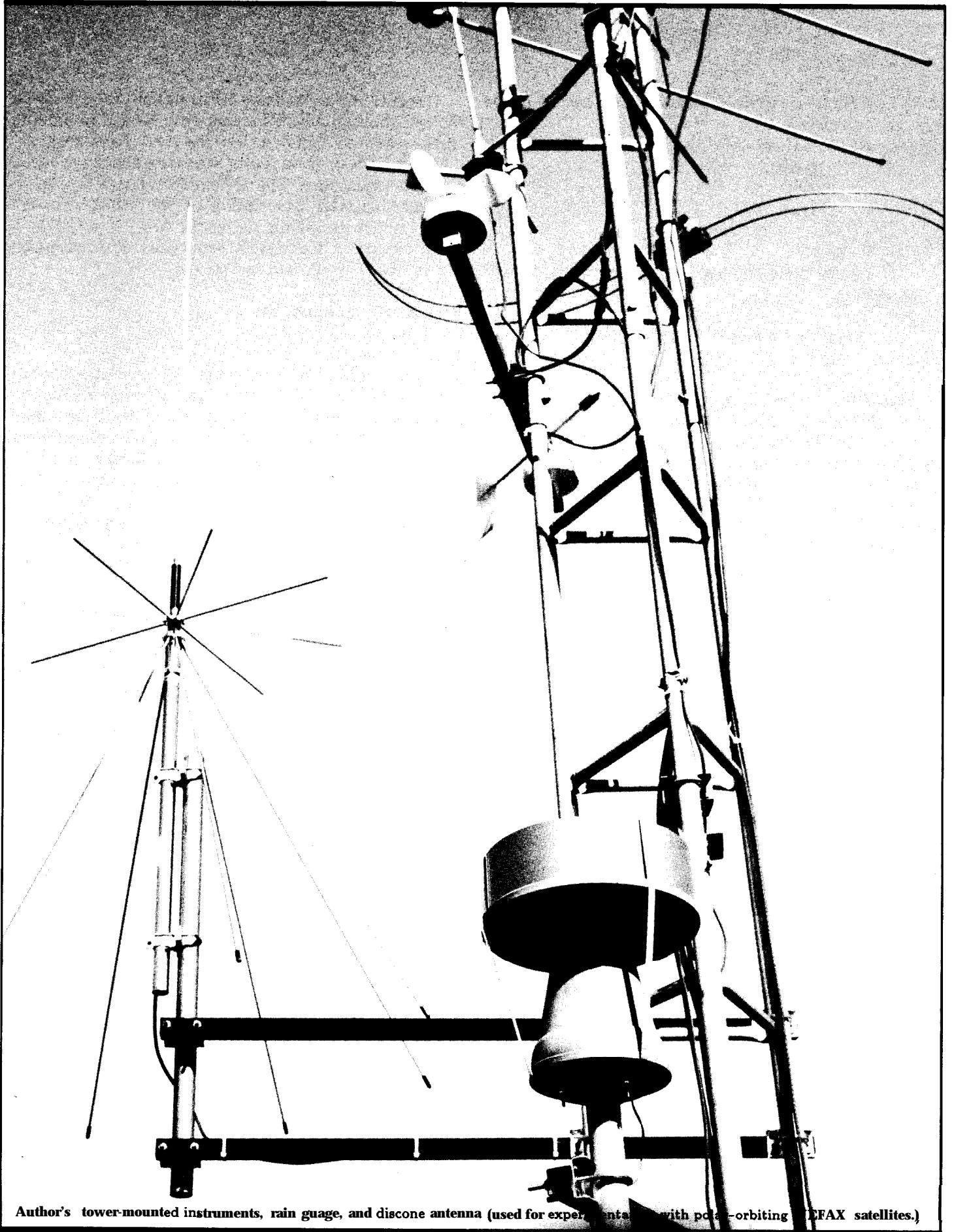
Most of the attention of the magazines goes to these operators, and the rest of us get left out.

The HPIB (aka IEEE-488) has never been our favorite parallel interface, although its uses with an array of laboratory instruments certainly validates its existence. Several companies make or have made specialized LSI ICs for HPIB control, among them Motorola, TI, and NEC. The Motorola 48488 is one of these chips (listed in a recent JDR Microdevices catalog at \$12.95.)

The specifications and applications data sheets for these chips are probably your best source of specific hardware design information. These sheets can be obtained from the literature departments of the manufacturers or, more quickly, from local industrial suppliers and distributors listed in the Yellow pages.

--INK ■

In Visible Ink, the Circuit Cellar Research Staff answers microcomputing questions from the readership. The most 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.



Author's tower-mounted instruments, rain gauge, and discone antenna (used for experiments with polar-orbiting E-FAK satellites.)

The Home Satellite Weather Center

by Mark Voorhees

Part 3

Weather Databases and System Software Overview

In this issue, we'll begin our tour of the Weather Data Processing System (WDPS) software, which will be used in an IBM PC or PC-compatible to handle several functions relating to the Home Weather Center we are building. Before that discussion, however, let's take a look at some of the database information available to you via modem (and supported by the WDPS software).

In its final configuration, your Home Weather Center hardware will provide the PC with many different types of information relating to local conditions, as well as satellite facsimile pictures of the world depicting weather system activity and movement. This data, however, is only a part of the total spectrum of information available to you. It would be impractical for our PCs to have direct access to every National Weather Service reporting station; yet regional, state, and national observations, radar data, and so on, are necessary if we are to have an overall view of current and future weather phenomenon. That's where access to a well-organized, accurate database is important.

I'll be discussing the capabilities of three major services, Accu-Data, CompuServe, and Weatherbrief, as well as the NOAA Bulletin Board and the Weatherstream 2000 system (a satellite-fed, continually updated data distribution system).

The Accu-Data Complete Computerized Database

Accu-Data is a service of Accu-Weather Inc., a major supplier of weather information and consulting services to businesses and news organizations across the country. You may be familiar with Accu-Weather reports on your local radio or television stations.

Using data from the National Weather Service, the FAA, military, and international sources, a staff of highly qualified meteorologists interprets weather conditions and trends to generate many custom products for users. Raw data from these sources is also available on the database, as well as text-format and high-resolution maps. The user can define his or her preference as to the format of accessed data. Some of the available data include:

-Hourly Surface Data -- as reported by the National Weather Service and other reporting stations

-Digitized Radar Information -- grid-related radar data from reporting stations across the country (the WDPS software includes a processor to build map graphics from the radar reports)

-Plain-Language Forecasts and Reports -- translated from various observation information by the National Weather Service

-Watches and Warnings -- from the National Weather Service

-Local, State, and National Summaries -- from the National Weather Service

-Complete Worldwide Reports -- using data from all sources

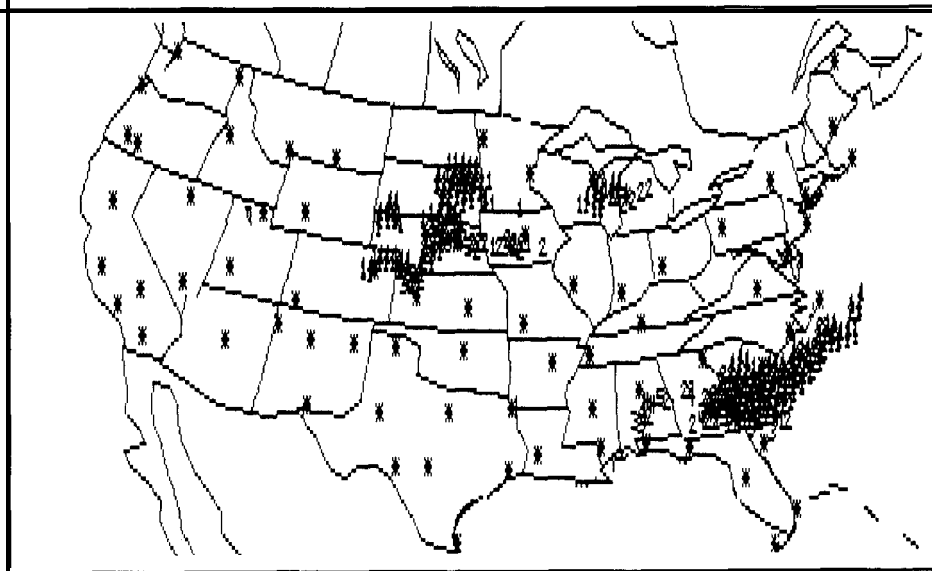
-Numerical Models -- for those individuals needing trend and other information in this format

-Upper Air Observations -- winds aloft, etc.

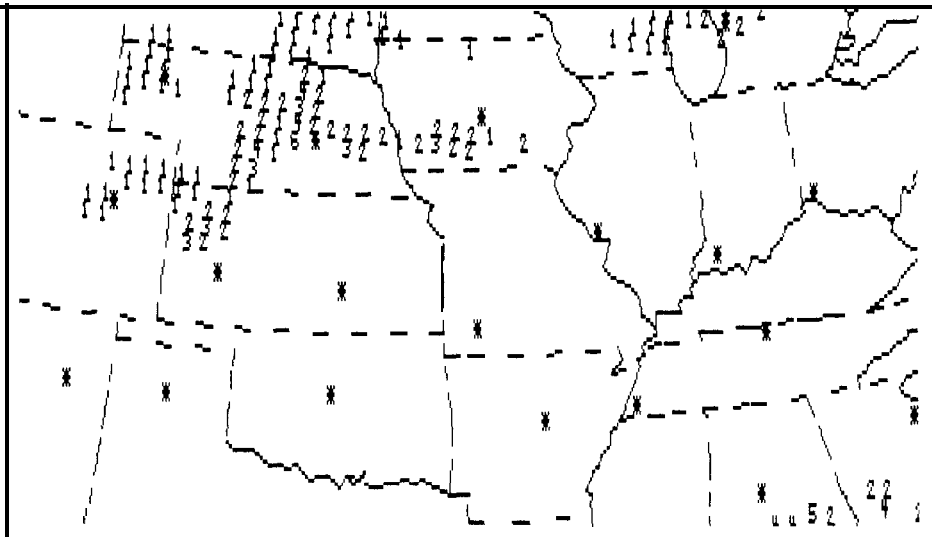
-Climatological Summaries and Normals -- for planning and review

The list goes on, but I think you can see that this is an all-encompassing system. Information is updated continuously, and you always have access to the National Weather Service and FAA high-speed data circuits in real time.

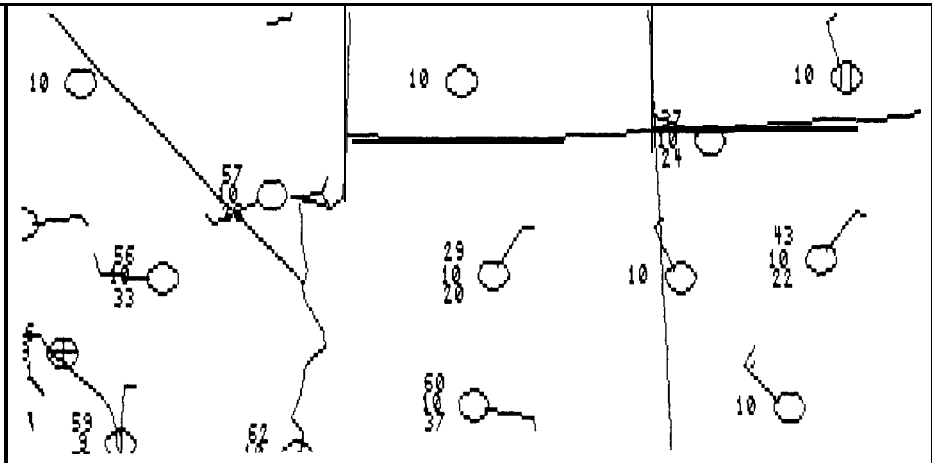
A major benefit of Accu-Data is the AMPS (Advanced Map Plotting System) high-resolution graphics, providing surface, radar, upper air, and computer forecast maps to Tektronix 401x-compatible terminals and communications software. (Version 1 of the WDPS access module for Accu-Data does not support this emulation, primarily due to difficulties in acquiring the protocol information from Tektronix. Version 2 of this module



composite radar map of the U.S.. Shaded areas are numbers indicating the intensity in a range from 1 (weakest) to 9 (most severe). The "*" symbol indicates the locations reporting clear conditions. The radar display software for the WDPS will use colors to depict levels.



composite radar depiction showing the numerical-type intensity markers more clearly. The MPS format allows customized area maps to be built by the user.



computer-generated forecast map for the Southwest. Besides showing temperature (top figure), visibility in miles (middle figure), and dewpoint (bottom figure), each station is represented with a symbol of a circle (open indicating clear, lines within indicating amount of cloud cover), with an optional line indicating wind direction, with flag lines at the end of the line indicating speed (each half-line is 5 MPH, each whole line is 10 MPH).

is planned for the purpose of incorporating this capability.)

Professional users are charged on a rate structure based on access rate and access time commitments. Although one of the most reasonably priced services in the industry, a client committed to a minimum of three hours per month can expect to pay at least \$180 in monthly access fees. Fortunately, Accu-Weather believes in supporting interest by hobbyists and weather enthusiasts, and gives us a very preferred rate of \$16.95 an hour, with a one-hour monthly minimum, for 300- and 1200-bps service. This rate allows normal user access and benefits with one exception: professional users have access to Accu-Weather's 800 access number; hobbyist users must pay their own long-distance charges (since access is available 24 hours a day, use of a long-distance service during discount hours will certainly save money).

I must stress that this lower rate applies to nonprofessional, hobbyist usage only, and is offered as a courtesy by Accu-Weather. User support from Accu-Weather is far above the norm for dial-up databases. I wrote a letter requesting information on the service, and not only did I receive an information package in a very timely manner, I was contacted by Regional Sales Manager Lee Gottschall, who was able to answer all of the questions I had concerning the service, as well as handle my service order. In all, this company demonstrates that it is a class act.

The user manual furnished for Accu-Data is also quite complete. In its 150+ pages, Accu-Weather covers everything from beginning user information to methods of acquisition of any data product. An extensive on-line help system is provided on the system, as well as a very detailed, user-interactive demo routine, but a good reading of the manual alone should make you

a very competent user.

I also had the chance to meet (via phone) Customer Service Representative Deb Kunce, who briefed me as to how to sign on for the first time, gave me my account number (the user assigns his or her own password), and made sure that their records were complete as to the equipment and software I was using (DEC VT10x emulation is fine for text, but the Tektronix emulation is required for AMPS displays). In addition to the ability to contact her with use questions and problems, I also have the ability to contact a computer operator at Accu-Weather 24 hours a day if I have a problem that can't wait. I must confess that I'm not used to this quality of service from any database service! I certainly can recommend Accu-Data as an excellent choice for on-line information. If you would like to inquire about Accu-Data service, contact:

Accu-Weather, Inc.
Attn: Lee Gottschall
619 West College Avenue
State College, PA 16801
(814) 234-9601

CompuServe

Anyone who's been involved in telecomputing is familiar with the 'household name' CompuServe. They support many weather database functions as a part of their Aviation Special Interest Group (AVSIG). Two different menu systems are available, designated as Public Weather and Aviation Weather. While not as complete as a dedicated database service, CompuServe's weather section does provide several of the National Weather Service products, including:

-Grid-Referenced Radar Observations (AVIATION)
-Winds Aloft (AVIATION)

-Hourly Reports (AVIATION)
-Terminal Forecasts (AVIATION)
-Local, State, and Extended Forecasts (PUBLIC)
-Weather Summaries (PUBLIC)
-Marine Forecasts (PUBLIC)

Information is updated approximately every hour. No real-time access to the weather wires is offered. I will also add that I've noticed an irritating failing in some of these CompuServe products. The information seems to be "edited" relative to that available on the wire services directly; that is, I've noticed that some reporting stations are not recognized within the CompuServe environment of some products. Not a major problem, but one worth noting.

As I mentioned in a previous article, the WDPS software is designed to support CompuServe's Graphics Interchange Format (GIF) for all graphics images. There has been a plan underway by the AVSIG personnel to generate GIF-encoded high-resolution maps automatically for on-line retrieval and downloading. As of this writing, that system has just been introduced, and I've had little chance to experiment with it.

If the products perform as promised, I plan to make any necessary module changes to support the function, and will include those changes in a Version 2 of the CompuServe access module. Normal CompuServe charges apply when accessing the PUBLIC weather area, however there is a nominal additional charge (\$0.25 per session) for access to the AVIATION products. A favorable benefit of CompuServe is, of course, the fact that most cities have access to a CompuServe Network node without incurring long-distance phone charges. User support is typical of that of other CompuServe services. No manual is provided, and little help is provided for the novice user. The best way to get questions

answered is to Email them to the sysop on-line. AVSIG does provide some weather-related public-domain programs in their data library, and CompuServe itself offers a personal color radar program for sale! but the program is quite basic in nature. Our WDPS radar display module will perform the same functions with greater detail and accuracy.

The CompuServe offering is certainly a reasonably priced alternative to a subscription to a dedicated database, keeping in mind its limitations. For more information on subscription to CompuServe, contact:

CompuServe Information Services
5000 Arlington Centre Boulevard
P.O. Box 20212
Columbus, OH 43220
(800) 848-8199

Weatherbrief

Weatherbrief is a service of Weatherbank Inc., a meteorological service company which is climbing in popularity among business users and news services. The service is available to hobbyists and enthusiasts at the regular user pricing. Weatherbrief uses data from the National Weather Service wires, the FAA, and other sources as a basis for their database. Weatherbank's meteorologists also create custom products using this information, including ASCII text-style graphic maps for each of three regions: West, Northeast, and Southeast. Among the text-style maps available are:

-Radar -- current and 3-hour-old reports
-Precipitation -- 12-hr, 12-24-hr, and 24-36-hr totals
-Sunshine
-Lightning Strike -- Western U.S. only
-Probability of Rain

-Recreation

- Long-Range Forecasts** -- lo-day, 30-day, and 90-day forecast maps
- Agricultural Soil Temperatures**
- Temperature Maps** -- Highs, Lows

These products are, of course, in addition to the raw weather before logging on, hence it does not service data products. Weatherbrief has promised a high-resolution graphics service also, and will provide users the appropriate PC software free of charge. As yet, however, the software hasn't been released, so I can't give you a report on its effectiveness. I will monitor the progress of this project and pass any further information on to you in a future issue.

Pricing for Weatherbrief is competitive, with the on-line **The NOAA Public Database**

charge set at **\$0.15/minute** (\$9 per hour) any time of the day, not including your long-distance charges. Their method of billing is unusual, however. Your initial fee of \$35 is placed in your "account," and your usage time charge is deducted from that amount. When your "account" gets to a minimal level, you are reminded to send payment to build your account balance. The system does provide an on-line summary of your usage for the current month or any of the past twelve months (you do not accrue on-line charges while accessing the summary).

My inquiry about the service was answered promptly by Steven A. Root, CCM, who is Vice-President of Marketing and Development. The user's manual was included with the reply, as well as an invitation for a demo period. I regret that this was the sum-total of Weatherbank's contact with me, and I have yet to get any other acknowledgement of my interest, even though I did send a \$50 check to begin service to my account. I sincerely hope this is not a precursor of things to come for the hobbyist/enthusiast user.

User support literature consists of a simple, seven-page manual, designed to give you basic system information. Weatherbrief depends on its on-line help subsystem to answer operational questions and makes the help system available to you even before logging on, hence it does not add to your access charges. Based on the products and pricing, Weatherbrief is certainly worth a look. To get more info, write or call:

Weatherbank, Inc.
ATTN: Steven A. Root, CCM
2185 South 3600 West
Salt Lake City, UT 84119
(801) 973-3131

The National Oceanographic and Atmospheric Administration (NOAA) makes some National Weather Service products available on a public database. This is, of course, not full NWS access as you would have from the dedicated database services, but it does provide some very unusual products:

-Climate Rankings

-Agricultural Products -- such as weekly growing degree-days for certain crops

-Heating and Cooling Degree Days

-Medium- and Long-Range Forecasting -- 5-day, 6-10-day, month, and seasonal

-Monthly Precipitation and Temperature Data

-Projected Palmer Drought Index

-Monthly Summary of Significant Climate Events

There are several more products and subproducts to choose from. The service is text only, and is available 24 hours a day. Various update schedules apply, depending on the product; the help listing does provide the update information for each product.

The major benefit of this service is its cost. There are no access fees, just your long-distance tolls. I've found that this service is quite handy when I'm looking for an unusual piece of information, but it obviously is not something you'll use every day.

User support is virtually non-existent except for the on-line help system and the ability to call one of three individuals at NOAA if you have a question. I suspect that this public service is a byproduct of a database that NOAA has created for its own use, so I'm not surprised at the minimal support. I should mention, by the way, that the system is so simple to use that I've never had to call the NOAA people, and I rarely find myself consulting the help screens. This service could be a convenient addition to your Home Weather Center, depending on your level of interest (the price is certainly fair). If you want to register as a user and acquire a password, call:

Mr. Fulwood, Mrs. Dionne, or
Mr. Patterson
NOAA (301) 763-4670

A Bulletin Board System of Note

Although this is not a weather database, and is not supported by the WDPS software, some of you will probably be interested in the information and programs available on the Datalink RBBS system in Dallas, TX ((214) 394-7438). The system is operated by the Dallas Remote Imaging Group and is dedicated to Weather Facsimile techniques and satellite- and Amateur-Radio-related topics. While the system does require a minimal subscription (\$18 annual membership) for full access, it does offer a very large library of user-uploaded programs, and a very active mail system.

Weatherstream 2000

I know I've only covered a few possibilities here, but I think you get an idea of what's available to your Home Weather Center in terms of National Weather Service data and custom weather database products. This is by no means a comprehensive listing. These services were chosen with an intent of maximizing performance while minimizing costs. For the interest of those individuals who would prefer (and can afford) full-time service, I want to mention one other service: Weatherstream 2000.

Weatherstream 2000 is a service of Satellite Information Services Corporation (SISCorp), which is primarily an information carrier service. SISCorp specializes in data delivery via satellite, using "mini earth stations" -- C-Band systems capable of reception using a dish as small as 24". The technology is quite sophisticated. The lower sensitivity of the small dish is offset by special techniques in RF transmission, multiplexing, and advanced error correction.

The current problem in the use of this type of service by individuals is the cost of the equipment to receive the signals. The "mini earth station" can cost \$3000 or more. National Weather Service products, Accu-Weather products, and FAA data is featured on the system. The service uses a hardware data selector to pass only the information that the user requests. Monthly fees are assessed based on the products se-

lected plus communication fees for the satellite time.

In all, while this service is quite complete, cost makes it practical only for commercial users. I am, however, investigating the possibility of creating a project which would be an alternative to the "mini earth station" purchase, which, if approved by SISCorp, could greatly reduce the initial investment and make this a

program and modules necessary to access these databases via modem. Before I begin, however, a short explanation as to how software will be handled in this series is in order.

I've chosen the "C" programming language for this project series to allow maximum processing speed, ease of portability to other systems, and the ability for the end user to customize and expand on my



Author's 1691 MHz Reception Hardware for WEFAX Satellite signals.

Weatherproof housing contains GaAs FET preamplifier and 1691-137.5 MHz downconverter.

viable addition to the Home Weather Center. I'll keep you informed of progress in this area. If you would like further information on Weatherstream 2000, contact:

Satellite Information Systems
Corporation
Attn: Paul Simmons
1100 Wayne Avenue, Suite 1209
Silver Spring, MD 20910
(800) 345-7472

General Software Notes

Now that we've investigated the possibilities for our database access, we'll turn to the Weather Data Processing System and highlight the main

processes as he or she desires. The finished programs will be available on the Circuit Cellar BBS, as well as directly from me for a minimal copying charge. Source code will also be available (see sidebar).

The programs will be presented in modular form. The main program, presented in this issue, is fairly generic. It calls the applications as "overlay"-type routines. In this way, we can minimize memory usage, increase speed, and create a simplified software interface.

The modules will be presented as they apply to the issue's discussion or project. Modules not present when called by the main program will simply display an "unavailable" message and return you to the

main menu. If you plan to use the requirements and recommendations Source code to make changes or add routines, you will need:

-Borland TURBO C, Version 1.5

-Window BOSS Revision 08.22.87 (a shareware product, available in small-model on CompuServe, or complete from Star Guidance Consulting, 273 Windy Drive, Water-

- IBM PC, XT, AT, or compatible computer
- At least 448K of RAM
- At least one 360K floppy disk drive
- EGA-compatible video card
- Hard disk drive recommended
- Hayes-compatible internal or external 300-, 1200-, or 2400-bps The

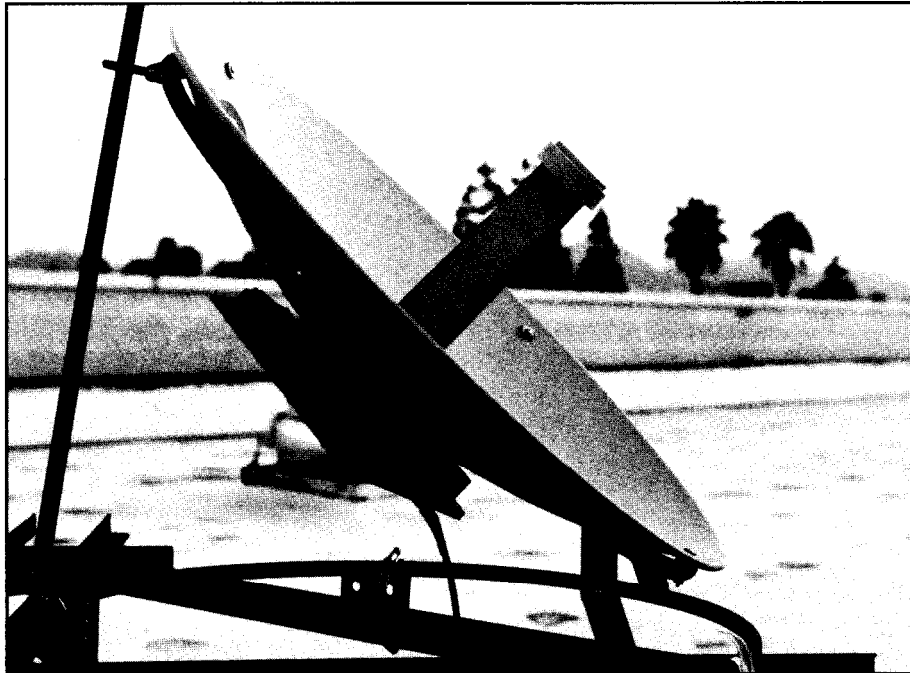
The Weather Data Processing System Software

The initial program to be presented is WEATHER.EXE, which is the main menu and configuration support for the WDPS software. As mentioned earlier, this program is the base of operations for WDPS. The main-menu screen will give you an idea of the scope of the total system -- from database communications and local instrumentation to facsimile support.

The only resident routine in the program, besides main-menu processing, is configuration entry and edit, where you can define your serial and parallel ports, instrumentation, and database user information. On first execution, the configuration file (CONFIG.WX) does not exist, so you are routed directly to the **configuration routine**. On the first screen, you will enter port numbers, modem speed, and instrument type. Follow the screen instructions; you will select this information from menus of possible responses:

- Modem Speed: 300, 1200, 2400
- Modem Port: COM 1, 2, 3, or 4
- Instrument Port: COM 1, 2, 3, or 4
- Printer Port: LPT 1, 2, or 3
- Instrument Type: Heath ID-4001, ID-5001

Please note that the instrument type selection is not meant to **indicate** that these are the only instruments to be supported. When I



"Mini earth station," C-Band satellite receiver, similar to type used for SISCorp's Weatherstream 2000 database service. Dish is only 24" in diameter, and can be successfully installed even in areas of high terrestrial interference.

bury, CT, 06705. Registration fee of \$50.00 provides all source code, telephone support, and free updates for one year.)

-LITECOMM Communications for the Home Weather Center) **Toolbox for Turbo C, Version 2.2** (a shareware product, available from Information Technology, P.O. Box 554, Coventry, RI, 02816. Fee is \$25.00 for the library alone, \$50.00 with source code.)

The WDPS software makes extensive use of windowing to speed up menu access. Additionally, the Communications Toolbox simplifies port and interrupt handling, and functions compatible with the Hayes instruction set. Hardware re-

modem on COM port 1, 2, 3, or 4 (to access databases)

- Available additional COM port (to access the 68000-based peripheral processor which is the hardware base for the Home Weather Center)
- Epson FX-series or compatible printer

There's no unusual equipment involved, and for most PCs or compatibles, this grouping is standard equipment. As always, comments, questions, and suggestions for future capabilities are welcomed. You can write to me at the address shown in the sidebar, or send **Email** to me on the Circuit Cellar BBS, the Datalink RBBS mentioned above, or on CompuServe (70566,777).

discuss instrument alternatives later in the series, I'll present some interface ideas for other devices, which will use the ID-4001's parallel interface format (the ID-5001 communicates serially), I do **recommend** the purchase of either of these devices for instrumentation since they appear to perform with reasonable precision over long-term use, require little maintenance, and are priced competitively with mechanical instruments having the same capabilities (specialized, commercially available electronic instrumentation is quite a bit more expensive!).

The screens that follow the hardware configuration allow you to enter the user name, password, and telephone access number for weather databases that you subscribe to. The communications modules will use this information for automated **signon**. If you don't **subscribe** to a given service, leave the screen blank. You'll be reminded that no account exists if you **should** attempt to access the **associated** module from the main menu. The main menu is self-explanatory. Entry of the appropriate line number will execute the associated module if that module is present in the default drive. This issue's software package also includes the first five application modules:

-PRINT.OVL -- a print routine to produce hardcopy of sessions captured by the communications modules

-CAD.OVL -- the Accu-Data communications module

-CCS.OVL -- the CompuServe 4WX communications module

-CNO.OVL -- the NOAA BBS communications module

-CWB.OVL -- the Weatherbrief communications module

WDPS support is not provided for Weatherstream 2000 since its usage would require that the PC be running at all times with software operating in background to process the data. This requirement would be beyond the design intents of our system. The communications modules are specially written for each service due to differences in sign-on procedures, operation, and terminal emulation. To the user, they have much in common operationally: they provide for creation of command files for automated retrieval, automatically access and sign on to the service, perform automated retrieval, allow manual operation, automatically capture the entire session, and perform log-off chores.

Experiment with the database(s) you choose in the manual mode for now. This is the easiest way to get used to the operation of the systems and the command structure.

10 the Next Installment

In our next installment, I'll give some examples of automating the retrieval process using the command files, graphic radar map depictions using data from a database, a **mini-review** on construction, calibration, and setup of Heathkit's new ID-5001 Advanced Weather Computer, and the initial discussion of our next hardware project: the 68010-based peripheral processor. ■

As noted in this article and elsewhere in this magazine, software for Circuit Cellar INK projects is available on the Circuit Cellar BBS and can be downloaded free of charge. If you prefer more personal service and would like to receive the WDPS software on disk, I will send it to you for \$6.00 (\$5.00 copying/disk fee, \$1 .00 P&H). Source code will be handled in a manner similar to "shareware," in that you will send a registration fee of \$35.00. This payment will cover source for all modules in the system and all updates or fixes. Software will be released in module groups. You will receive the appropriate disk shortly after it is published.

To order software or source code on disk, send check or money order to:

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

Allow 30 days for shipment.

Graphics presented in this article are examples of Accu-Data AMPS-format maps, using commercially available communications software. The AMPS-format capability is planned for a future version of the Weather Data Processing System Accu-Data communication overlay.

Corrections

Vol. 1, No. 1: The Home Satellite Weather Center-Part 1: RGBI to NTSC Converter by Mark Voorhees:

Page 32 -- Figure 4 should be rotated 180 degrees

Page 33 -- Missing ground symbol from junction T1/C18/R16

Page 33 -- Label on IC5 pin 5 should be AI 3

Vol. 1, No. 2: The Home Satellite Weather Center-Part 2: NTSC Encoder Alignment and System Overview by Mark Voorhees:

Page 25 -- The fine-adjust of R11 & R12 should refer to Figure 1

EDITOR'S NOTE

Accu-Weather Forecaster

Right around the time that Mark's article showed up at the INK office, we obtained an evaluation copy of a new program called Accu-Weather Forecaster. It fit into the scheme of Mark's article so well that we thought you should know about it.

Developed by Metacom Software Inc. in conjunction with Accu-Weather Inc., Accu-Weather Forecaster is designed to be a user-friendly, automated front end to Accu-Weather's on-line Accu-Data information service.

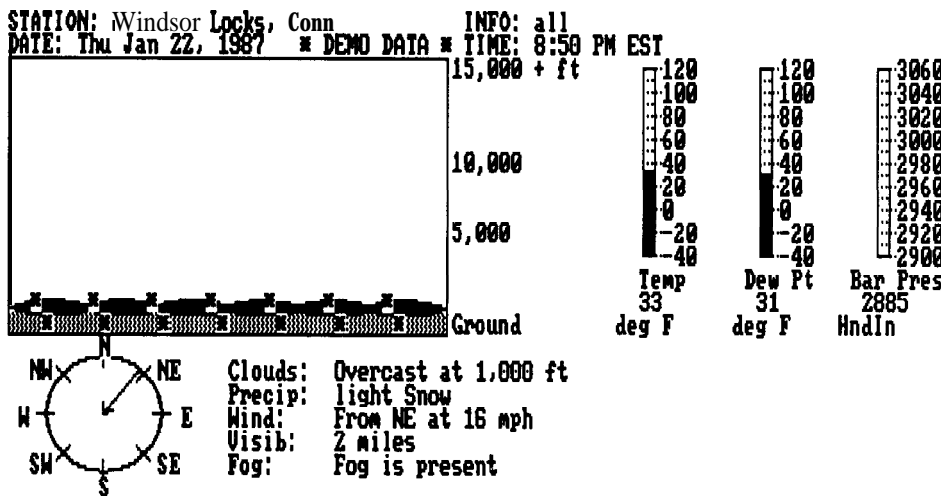
While off-line, the user tells the program just what weather-related information he wants to know. The program will then automatically call Accu-Data, download all the necessary information as quickly as possible, and terminate the call. The user is free to look over and analyze the data off-line for as long as he wants without incurring any additional connect charges.

A typical information request consists of current data from 100 major stations throughout the U.S., current data from every station within 250 miles of the user's home base, hourly data covering the past 24 hours from the user's local station, the user's local forecast, and the National Weather Service's analysis of current national weather conditions. Such a request typically takes about four minutes to download and costs about \$2.00.

The program uses maps, graphs, pictures, and shading to assist in analyzing the data, and several dot matrix and laser printers are supported to produce hardcopy. It is supplied with a very complete 172-page user's manual plus a 37-page "forecasting guide" designed to assist novice weather enthusiasts in making their own predictions.

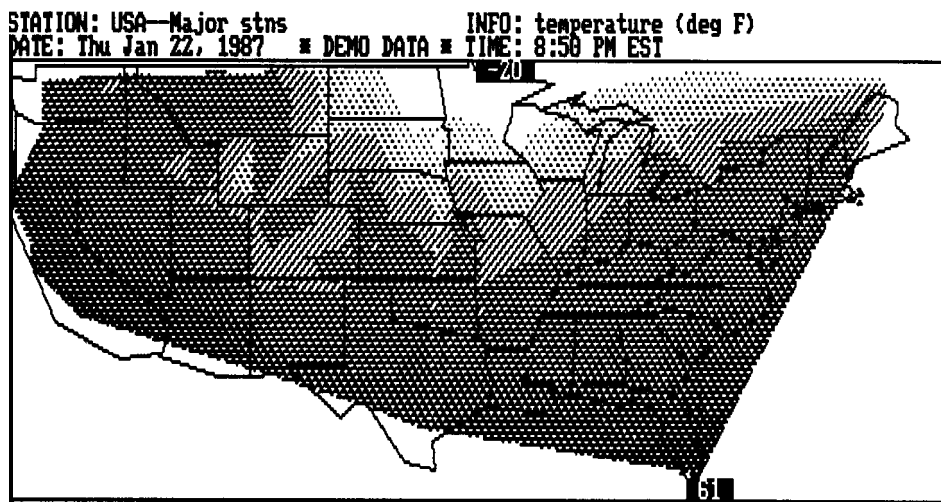
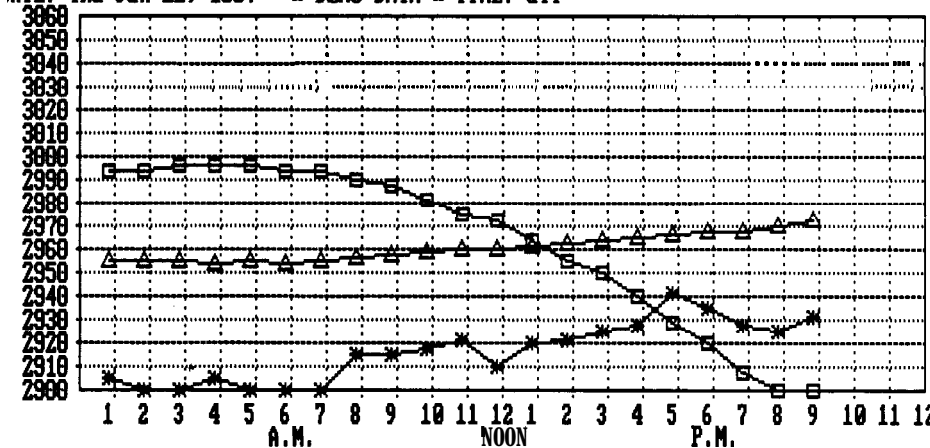
Accu-Weather Forecaster is available for an IBM PC or compatible with 256K of RAM and CGA display, or a Macintosh with 1 megabyte of memory. An Apple II version is said to be under development. The program costs \$89.95 and is available from Metacom Software Inc., P.O. Box 31337, Hartford, CT 06103; (203) 223-5911.

As a snow storm roared into New England on January 22, 1987, the data at the left was recorded by Accu-Weather. At top left, cloud cover and other information is shown pictorially. And the barometric pressure dropped quickly (middle left) as the temperature and wind speed rose slightly. Bottom left shows the temperatures across the country as different shades of grey.



- = barom pres
- △ = temperature
- * = wind speed

STATION: Windsor Locks, Conn INFO: barom pres (HndIn)
 DATE: Thu Jan 22, 1987 * DEMO DATA * TIME: all



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

Sysop: Ken Davidson

*Starting with this installment of **ConnectTime**, we're trying something new. In trying to pick out message threads for publication, I found that the most interesting ones were usually the longest ones. In order to pack more information into the space allotted to **ConnectTime**, we have gone to a smaller typeface. A discussion thread that took over seven pages using the standard typeface (and, hence, couldn't be published) can be condensed to fit into just four pages.*

Of course, having more information available is useless if you can't read it. I'd like to hear your opinion. Is having more information available in each issue worth the smaller typeface, or would you rather have less information that is easier to read? Drop me a note the next time you're on the Circuit Cellar BBS and we can discuss it.

Discussions that start off with an innocent question can often digress into something infinitely more interesting. The following discussion thread started with someone asking about a simple intrusion detector and ended up with a debate over the best placement of landmines in the front yard!

Msg#: 1163 *PROJECTS*
 From: ALAN GOLDSTEIN
 To: STEVE CIARCIA
 Subj: INFRARED BURGLAR ALARM

A friend just bought a house with a screened-in patio and would like some sort of break-in alarm. She has cats, so general motion detectors are out. Some sort of highly directional IR beam across the opening but very close to the screening seems best to me. Small packaging would be nice for aesthetics. Without searching through all the back issues of BYTE, you may have the answer off the top of your head. Any thoughts about kits, methods, etc.? Thanks

Msg#: 1186 *PROJECTS*
 From: STEVE CIARCIA
 To: ALAN GOLDSTEIN
 Subj: REPLY TO **MSG#** 1163 (INFRARED BURGLAR ALARM)

I did an article a while back that was a doorway sensor that

could tell the difference between people and animals. Also, I have IR detectors all over the place. My experience is that they work fine in daylight and my little Westie triggers all of them. I believe you were talking about protecting a door screen. In my house, at least, the screened cellar windows are protected by a thin wire stretched across them connecting two contacts. Anyone enters the cellar through the screen and they pull the stretched wire **from its** contacts. Seems to work fine.

-- Steve

Msg#: 1197 *PROJECTS*
 From: ALAN GOLDSTEIN
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1186 (INFRARED BURGLAR ALARM)

Thanks. I'd considered the wire approach, but thought it was too much of a one-shot idea (the occupant is not technical or handy and would have trouble restringing the wire). Also, it is primarily designed to wake a person and scare a trespasser, so a quick nonrepeating signal is all that's required.

Msg#: 1165 *PROJECTS*
 From: BOB PADDOCK
 To: ALAN GOLDSTEIN
 Subj: REPLY TO **MSG#** 1163 (INFRARED BURGLAR ALARM)

The March 17 issue of Electronic Design, on page 146, describes a component: "Infrared Sensor Responds with Digital Logic Signal." "When a person or body-temperature object moves into the field of view of the **IR1000** digital passive infrared-sensor module, the module responds with a digital-logic output signal. The module's sensitivity can be adjusted with the reference input, enabling the unit's range to be varied. Daylight operation is not a difficulty for the sensor because it responds to changes in infrared radiation from 8um to **14um**, the optimal range for atmospheric transmission of **95°F** (skin temperature) blackbody radiation. Contact Infrared Inc., PO Box 47, Parlin, NJ, 08859, (201) **721-7160**"

I don't know about cats, but a dog's body temperature is around **102°F**. Would it set the thing off?

Msg#: 1167 *PROJECTS*
 From: ALAN GOLDSTEIN
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1165 (INFRARED BURGLAR ALARM)

I was really aiming (no pun) for an active system with an emitter and photosensor, so that when someone broke the beam it would set off an alarm circuit. You do bring up a point: how immune is this type of system to sunlight? I guess a directional system would not be affected

by stray sunlight. The system would be high enough so that no animals could break the beam, but still provide enough coverage of the window area that a person could not get past it.

Msg#: 1205 *PROJECTS*

From: BOB PADDOCK
To: ALAN GOLDSTEIN
Subj: REPLY TO MSG# 1167 (INFRARED BURGLAR ALARM)

Since I'm overly cynical, I always assume someone is going to try to defeat the system. What prevents someone from going over or under the beam?

Msg#: 1256 *PROJECTS*

From: KEN DAVIDSON
To: BOB PADDOCK
Subj: REPLY TO **MSG#** 1205 (INFRARED BURGLAR ALARM)

The trick is to set up any sensor so a potential victim doesn't know it's there. It's pretty tough to avoid something you don't know about.

Msg#: 1268 *PROJECTS*

From: BOB PADDOCK
To: KEN DAVIDSON
Subj: REPLY TO **MSG#** 1255 (INFRARED BURGLAR ALARM)

Yes, I **realize** that. But I want to protect against someone like me who would take the trouble to carry IR sensing equipment, RF sniffers, nonlinear junction detectors, the whole nine yards. The dogs adequately protect against the average delinquent. If the system is set up to make it difficult for the professional, then your average **off-the-street** burglar will have an almost impossible time.

Msg#: 1288 *PROJECTS*

From: STEVE CIARCIA
To: BOB PADDOCK
Subj: REPLY TO MSG# 1268 (INFRARED BURGLAR ALARM)

You're sounding paranoid, Bob. :) Land mines in the front yard?

The beauty of passive systems is that they emit nothing to detect. Yes, they can be beaten, but an overlapping system using various technologies of surveillance is almost impossible. You can install an **IR** beam unit but it can be detected with a can of hair spray. Most crimes are not committed by professionals so unless you have a bank vault in your house or your address is next to a maximum security prison and you're the first house after they go over the wall, I'd stick to reasonable.

Of course, look who's talking. I have so much junk installed in my house it looks like the **U.S.** embassy after the KGB has been there. Recently I was asked what was so valuable that I had to install all this junk and my conclusion was that the most valuable stuff in the house was the alarm/surveillance system. :) I'm just having fun guys.

At least I can document my eccentricities and make money at it. :)
 --Steve

Msg#: 1292 *PROJECTS*

From: BOB PADDOCK
To: STEVE CIARCIA
Subj: REPLY TO **MSG#** 1288 (INFRARED BURGLAR ALARM)

If I sound paranoid, it is because I am. Wait till you read what

I'm going to post on April **1st**, then maybe you will see why I keep looking over my shoulder. "The Guardians of Status Quo" don't like people rocking the boat.

Msg#: 1172 *PROJECTS*

From: RICHARD ANDREWS
To: BOB PADDOCK
Subj: REPLY **TO MSG#** ii65 (iNFRARED BURGLAR ALARM)

Is IR really the way to go? What about using an ultrasonic transducer? Polaroid sells kits and I believe that Micromint has in the past also. There is now a "ruggedieed" version available from Polaroid which would take care of any weather considerations. If the transducer is mounted far enough above the floor, animals would not set it off. Good luck.

Msg#: 1191 *PROJECTS*

From: BOB PADDOCK
To: RICHARD ANDREWS
Subj: REPLY TO MSG# 1172 (INFRARED BURGLAR ALARM)

I really couldn't say which would be better. IR is generally simpler, unless you need to get into complicated optics. I would like to find a sensor that only responds to humans. I have a somewhat isolated house, with every type of animal running around outside, from chipmunks to full-size horses. I don't really care if there is a chipmunk on my front porch (the horse won't fit), but I want to know if there is "someone" out there. Also, ultrasonic is much easier to defeat than a modulated IR system if you are ever so inclined.

Msg#: 1193 *PROJECTS*

From: KEN DAVIDSON
To: ALAN GOLDSTEIN
Subj: REPLY TO **MSG#** 1163 (INFRARED BURGLAR ALARM)

The easiest solution to this is to go out to Radio Shack and get one of their units that sends out a pulsed IR beam to a reflector that gets bounced right back to the unit. When anything breaks the beam, the unit sounds its own alarm and also closes a relay contact. If mounted high enough above the floor, it could be immune to any animals since the beam is highly directional. Since it uses a pulsed beam, it **would be relatively immune to sunlight and attempts to defeat it. It also has enough range to reach across any typical room.**

Msg#: 1199 *PROJECTS*

From: ALAN GOLDSTEIN
To: KEN DAVIDSON
Subj: REPLY TO **MSG#** 1193 (iNFRARED BURGLAR ALARM)

Thanks, I guess I'll take a run to RS. Unfortunately, most of their stuff is usually bulkier than is called for (I've seen their IR sensor module in use at the store).

Msg#: 1253 *PROJECTS*

From: KEN DAVIDSON
To: ALAN GOLDSTEIN
Subj: REPLY TO **MSG#** 1199 (INFRARED BURGLAR ALARM)

Well, if you want something quick and dirty that's going to keep working, that's the unit you want, even if you think it's too big. In my opinion, it's just about the right size for what it does.



*Guest
Editorial*

The Information Free -For-All

by Mark Dahmke



SPOT

I have always assumed that someday the rate of growth of chip density would level off as certain fundamental limits of physics were reached. In 1976 a friend of mine confidently predicted that with electron beam lithography you couldn't go much beyond one megabit on a chip or, like the human brain, it would start to forget things. In a college physics class, we once ran through some calculations related to gate size, power consumption, quantum fluctuations in the flow of electrons, and switching time and concluded that you could reach a point where you could switch a transistor and no electrons would flow. In spite of these "obvious" physical limits, some Japanese chip manufacturers are predicting gigabit memory chips (to be used in everything from computers to toaster ovens). Regarding disk storage, one manufacturer recently announced the development of a two-inch loo-megabyte drive. Optical write-once technology will soon be reaching and surpassing one gigabyte per disk. What happened to the fundamental limits we all assumed were there? A few years ago I still would have guessed that we would reach a plateau, but now I'm not so sure.

If we consider the broad range of uses for both random-access storage devices and magnetic or optical storage, the consequences are both exciting and a little frightening. The recent debate over the sale of DAT (Digital Audio Tape) recorders is a good case in point. The speed with which the Video Cypher scrambling system was broken should be a guide to further legislation regarding anticopying schemes based on digital techniques. Most software vendors have removed copy protection schemes from their products, which is a good move for the industry as a whole. But as more and more intellectual property takes digital form instead of being distributed on paper, the problem of protection becomes more complicated. Even printed works are no longer safe. With a good OCR scanner, you can take a phone book, encyclopedia, dictionary, or any other large reference work and turn it into a machine-readable database at a relatively low cost. The high cost of rekeyboarding would have stopped most of this kind of plagiarism, but fast scanners are removing that barrier also.

In his hypertext concept, Ted Nelson describes a system called Humbers (huge numbers) as a way of recording the uniqueness and ownership of every byte or character in a document. When any document makes reference to or copies material from another hypertext document, that use will automatically be recorded and credit given to the original author. This system could one day solve the problem of plagiarism, but it plies a massive and wholly electronic/hypertext infrastructure or it won't work.

It's difficult to imagine the rate at which megabytes of information and software travel across this continent on a daily basis -- just through the use of Bulletin Board Systems alone. This highly democratic means of communication has taken everyone by surprise in the last ten years. Bulletin Board Systems, commercial time-sharing systems such as CompuServe and BIX, and database services like Dialog are rapidly breaking down many barriers to the flow of information, including national frontiers, but not, unfortunately, those separating rich and poor people.

As we have seen with video cassettes and audio tapes, the only thing that really stops unauthorized copying is holding the cost of the product down to less than the cost of making a copy. In the case of software, copying is limited by the need for support and updates after the purchase. For the case of property where one does not expect to receive ongoing support, the only apparent limitation to copying is cost of the copying medium (diskette, WORM disk, xerox copies, DAT, telecommunications).

In the next decade, we could see many responses to this technology, depending on how far copying costs drop, ranging from tighter controls and inconvenient (not to mention inadequate) copy protection schemes for just about everything, to an information free-for-all of the likes we've never seen before. The fuss over xerox copiers and the attempted controls over copying for "personal or educational" use would look trivial by comparison. For publishers, this will be a nightmare, and they will someday have to adapt to a market place driven as much by mass storage devices as by paper and ink. And what will it mean for the rest of us? Perhaps we will end up watching over our shoulder for the Copyright Police, or maybe the shareware concept will prevail and become the model for distribution of all intellectual property. ●

Msg#: 1203 *PROJECTS*
 From: STEVE CIARCIA
 To: ALAN GOLDSTEIN
 Subj: REPLY TO **MSG#** 1199 (INFRARED BURGLAR ALARM)

RCA markets some real tiny IR motion detectors. They are found in most discount stores in the lighting section.
 -- Steve

Msg#: 1254 *PROJECTS*
 From: KEN DAVIDSON
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1203 (INFRARED BURGLAR ALARM)

That's another good alternative. The RCA motion sensors have a **180-degree field** of view horizontally, but a very narrow field of view vertically. You could mount a sensor on the wall about four feet from the floor directly across from the windows. A dog walking into the room wouldn't trip it, but anyone coming in through a window or even walking into the room would trip the sensor. You can get a version that closes a relay contact for less than **\$50**. (It's the one meant for outdoor use.)

Msg#: 1202 *PROJECTS*
 From: STEVE CIARCIA
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1191 (INFRARED BURGLAR ALARM)

Ed Nisley and I have been trying to pair an OEM-286 (CCAT) with an **ImageWise** digitizer to produce, in effect, a "people identifier." So far, this "price-is-no-object" approach is too slow (even using a parallel digitizer) to be effective. What he needs is what I installed but have not written about because it is someone else's commercial equipment and I don't do reviews.

I installed a Sony Monitorscan system. It is a computerized unit with up to eight video cameras. On each camera's display, you designate "sensitive" areas with rectangular blocks (**10-90%** of the screen) that are used for motion detection. You can, for example, have a camera pointed at a **forest** path but only trigger motion above the **4.5-foot** level, or between six inches and two feet. You can precisely set the areas of motion detection! And, you can also set the degree of motion to trigger an alarm. The Monitorscan sits there looking at all these cameras and if any one has motion, that camera is switched to a live monitor output and it triggers a contact closure output for alarm, and, oh yeah, it has a built-in video printer and presents you with a picture of the perpetrator (horse?) in the subject picture. The alarm output can be used to turn on a video tape recorder that records the alarm camera. As a perpetrator walks around, into the field of the other cameras, the Monitorscan will automatically switch to those cameras.

Now the bad news: A Monitorscan and eight cameras (the cameras are about the size of a cigarette pack) is about **\$10,000**. There are some other neat toys you can add like time-lapse video recorders, etc. :-)

Ed and I have been trying to duplicate this unit as a project but no luck yet. Call a Sony Surveillance Systems dealer and get some info on Monitorscan. It has its faults but it's neat.
 -- Steve

Msg#: 1207 *PROJECTS*
 From: BOB PADDOCK
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1202 (INFRARED BURGLAR ALARM)

I don't find motion detectors particularly useful. If one is looking at a garden path, what happens when the wind blows? A bush

moves, a dead tree limb falls to the ground, etc.? Most alarms (and sensors) seem to be set up for cities, where you have very structured access points (roads, lobbies, hallways, etc.). The ultimate would be to be able to read the mind of the approaching person to find out if he has hostile or detrimental intentions toward myself or my possessions. Alas, that is a way's away (maybe tomorrow?).

Msg#: 1241 *PROJECTS*
 From: STEVE CIARCIA
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1207 (INFRARED BURGLAR ALARM)

Remember, Bob, I've got them all over the place so I'm speaking from experience. Microwave motion detectors are the most sensitive. They pick up movement of anything "conductive." Unfortunately, trees, especially wet trees, and rain drive it nuts. Forget microwave outside. Passive infrared like that used in those outside lighting units sold with two flood lights works quite well. I've had virtually no false alarms except in the really harsh weather. Since trees are the same temperature as the surroundings, it ignores them. Unfortunately, it will pick up dogs.

I think you have to decide what you want to do with the info provided by the IR sensor. If it is an alarm system then it has to have ironclad response or it's useless. I use all the outside sensors as "warning" systems. Lights go on sequentially and anyone approaching the house definitely knows that "something" knows they are there. I agree that reading the person's mind might allow a more direct preemptive strike, but legal authorities sometimes frown on it. Save the land mines for inside the house where the legality is less in question and go for deterrence outside. :-)

-- Steve

Msg#: 1361 *PROJECTS*
 From: RICHARD ANDREWS
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1205 (INFRARED BURGLAR ALARM)

What's **to prevent someone from** taking a chainsaw and cutting their own doorway through one **of your** walls? At some point you've got to say that it's close enough.

As any driver in the Boston area (U.S. car theft capital) will tell you, if somebody want what you've got badly enough they will find a way to steal it.

Have a nice day.

Msg#: 1312 *PROJECTS*
 From: BOB PADDOCK
 To: RICHARD ANDREWS
 Subj: REPLY TO **MSG#** 1301 (INFRARED BURGLAR ALARM)

Amazing. I've gone from wanting a simple burglar alarm that was insensitive to weather-induced motion and a large range of animals (chipmunk size to horse size) to landmines and cannons in the driveway.

Actually, the most valuable (i.e., hard-to-replace) stuff I possess is information: lots of books, magazines, etc. The burglar might try to carry off my microwave or, if they're into pain, the 50-year-old fridge, but I doubt they're going to steal my library.

On the other hand, there are those (the "Big Brother" types) that consider knowledge to be very dangerous when it doesn't fit into "their" scheme of things. A fire would be my worst fear come to life, and they don't even need a chainsaw. **They** could just drop some napalm from a passing airplane. Of course, I have hidden away copies, or gotten fire-proof boxes, **halon** fire extinguishers, and so on. Like computer programs, you always need backups.

Msg#: 1327 *PROJECTS*
 From: STEVE CIARCIA
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1292 (INFRARED BURGLAR ALARM)

Bob, perhaps you are trying to do too much with one kind of sensor or alarm system. I have approached security with a multimode/multitech scheme that you might want to consider. There are three components: warning/lighting, detection/alarm, and evidence. Let me explain.

An HCS controls convenience and perimeter lighting in my home. Anyone approaches, we don't incinerate them. The HCS knows someone is outside (while the alarm is set inside) and it turns on a few high-power floodlights and sequences a few lights inside. Politely, it says, "I know you are here!"

Next is the real hard-wired alarm system. Perimeter switches on sliding glass door and swing doors, and motion detectors for inside living areas (catches the random chainsaw-entry burglar). Any break in the perimeter or a motion detector and it calls the police (silently).

Finally, there is a surveillance system (actually I was just playing with video and it was a convenient application) with assorted cameras inside (looking at the equipment in the entertainment room) and outside. As I mentioned before, this system is auto motion detecting and includes a time-lapse recorder and video printer.

All three systems are independent but they do share some cross-connection of sensors (**optoisolated** connections so one system can't trash the other) for inside and outside motion.

So what does all this do? Nothing if the world is honest; a lot if they are dishonest. I have constructed this multimode system because you are ultimately alone when it comes to your security. I've seen too many cases where the perpetrators are never caught, or the police lack the attention or intelligence to do investigative work. If I can't scare unwanted people away by the automatic lighting then we have some obviously nasty people to deal with. If they perpetrate a crime then the police will be involved. When the police say, "a lot of houses have been hit lately. Too bad we missed him.," you can say, "Well, here's his picture and a picture of the car he was driving."

No, you can't stop the professional. But it is the amateur that will do the most damage and is the hardest to catch (no record, no evidence). Think in terms of levels of protection and don't try to do it all with one sensor.
 --Steve

P.S. I discovered through six months of surveillance system use that fewer shady characters visit my house when I'm gone than I thought. Also, I presented (polite for "bagged") UPS with a complaint about backing their truck over my shrubs, complete with pictures. :) And finally, neighbors are funny. When I was at CES in January my neighbor (supposedly) plowed my driveway. I thanked him, asked how long his **14-year-old** kid had been driving his truck, and gave him a picture of the event! What do I add for the fourth level? Laser targeting?

Msg#: 1340 *PROJECTS*
 From: BOB PADDOCK
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1327 (INFRARED BURGLAR ALARM)

Wouldn't a professional thief take the time to disrupt power and phone communications? You don't have to get within range of anything to do that in most places. Around here the police won't accept calls from anything but a live person.

Is laser targeting an upgrade of "I've Got You In My Scanner"? The problem then becomes what do you do (that will keep you out of

jail) to the "Locked-on Target"?

Msg#: 1363 *PROJECTS*
 From: STEVE CIARCIA
 To: BOB PADDOCK
 Subj: REPLY TO **MSG#** 1340 (INFRARED BURGLAR ALARM)

Yes, a professional burglar might cut the phones and power. That's why there are sirens OUTSIDE just below the eaves. They accidentally went off once when I smoked a pork chop on the kitchen grill (the fire alarm is part of the security alarm) and they are loud. Turned that sucker off fast. Remember: three systems, all independent. HCS and lighting are all AC powered. Cut the lines and it's over. The alarm system has its own batteries. If the phones are out then it uses audible alarms. The surveillance system is totally UPS backed, including the cameras. But, you say how can you take pictures in the dark down in the Circuit Cellar? The automatic battery-backed lighting, of course (low voltage). :-)

And if all else fails, did I mention the la-gauge in the closet? :-)
 -- Steve

Msg#: 1348 *PROJECTS*
 From: ALLAN LONG
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1288 (INFRARED BURGLAR ALARM)

Regarding the question of what do you do with the locked-on target: high-intensity sound or light. What about some kind of system to aim a video camera?

Msg#: 1361 *PROJECTS*
 From: STEVE CIARCIA
 To: ALLAN LONG
 Subj: REPLY TO **MSG#** 1348 (INFRARED BURGLAR ALARM)

That's what Ed Nisley and I have been working on all along! **ImageWise** was designed so that we could **digitize** a picture and point a camera. While I was at CES in January I bought two pan-and-tilt camera mounts. We're getting there.
 -- Steve

Msg#: 1380 *PROJECTS*
 From: ALLAN LONG
 To: STEVE CIARCIA
 Subj: REPLY TO **MSG#** 1361 (INFRARED BURGLAR ALARM)

What about a starlight-style image intensifier to pick up the little buggers without visible lights? I understand that a single far-red LED will be sufficient for a visible image at around 15 feet. Where to get a plate (used to prevent blooming) would be a problem. What do you think? A great INK project or what?

Msg#: 1416 *PROJECTS*
 From: STEVE CIARCIA
 To: ALLAN LONG
 Subj: REPLY TO **MSG#** 1380 (INFRARED BURGLAR ALARM)

I said workable technology, not off the wall. Yeah, the surplus catalogs are full of night-vision scopes, but it is easier to turn on the lights. I have a low-light-level TV camera in the driveway that is really

great but all this low-light stuff is real expensive.
-- Steve

Msg#: 1430 *PROJECTS*

From: **ALLAN LONG**
To: STEVE CIARCIA
Subj: REPLY TO **MSG# 1416 (INFRARED BURGLAR ALARM)**

I **kinda** figured it might be expensive. Thought maybe something interesting could be done with the **ImageWise** digitizer. Alas, back to the old thinking cap.

Most people have seen laser shows done in planetariums or rock concerts where laser light is used to create patterns, objects, and even words on walls and ceilings. The following discussion focuses on one effort to make a homemade laser deflection system.

Msg#: 1149 *GENERAL*

From: DALE NASSAR
To: ALL
Subj: LASER WRITING

I am trying to find an easy way to use a He-Ne laser to write by projecting the beam on a screen. The electronics are no problem but I need help with the mechanics for deflecting the beam. I first tried mirrors attached to speakers but there was too much distortion due to **IR** droop (inertia/resonance). Servo motors were too slow (I haven't tried **steppers** yet). Does anyone have any ideas? The electronics I came up with are great for oscilloscope graphics, but that's about it. Thanks.
--Dale

Msg#: 1169 *GENERAL*

From: RICHARD ANDREWS
To: DALE NASSAR
Subj: REPLY TO **MSG# 1149 (LASER WRITING)**

One way to do this is to use a mirror that is deflected by a piezoelectric element. Of all the methods that I have seen to date this approach gives the best bandwidth, which will give you the fastest writing **speed**. Piezoelectric elements are crystal structures that distort when subjected to high voltages (**100+** Volts, generally). One company that I know of that specializes in this type of stuff is Burleigh; I believe they are in New York State someplace. They make lots of amazing stuff. If I recall correctly they sell a high-voltage op-amp box which takes a low-level analog voltage and amplifies it to the appropriate levels.

One other source might be the larger laser manufacturers such as Spectra Physics. Good luck.

Msg#: 1176 *GENERAL*

From: DALE NASSAR
To: RICHARD ANDREWS
Subj: REPLY TO **MSG# 1169 (LASER WRITING)**

Richard, thanks for the information about piezoelectric devices -- I'm going to try it. I also heard that galvanometers were good for this.
--Dale

Msg#: 1178 *GENERAL*

From: STEVE CIARCIA
To: DALE NASSAR
Subj: REPLY TO **MSG# 1149 (LASER WRITING)**

I've been trying to build a laser deflection system for a couple years. Speakers and other cheap mirror movers aren't very good. Real X-Y deflection systems use galvanometer-driven mirrors (nonlinear drive, unfortunately). I visited the place where they are manufactured in Massachusetts hoping to buy one at a good price. The cheapest I could get away with was about \$4600 (yea, that's right!) for simple **X-Y** from a linear analog input. The galvanometer mirrors alone cost about \$500 each for the "**crummy**" ones. These guys have a lock on the market, unfortunately, because the technology is patented and there is a limited market. My latest approach is piezoelectric reflective plastic. I have some but I haven't tried it yet. Believe me. I have five lasers(!) and INK WILL have some laser articles! Got to do something with this junk. :-)

-- Steve

Msg#: 1181 *GENERAL*

From: DALE NASSAR
To: STEVE CIARCIA
Subj: REPLY TO **MSG# 1178 (LASER WRITING)**

Steve, I worked with laser deflection systems several years ago for a physics seminar and have only tried speakers with fairly large mirrors since they worked well with that application. I attempted to make it write after seeing some curves produced (on computer) by cubic spline because these computer-generated curves looked so much like the curves produced by the mirror system.

I am thinking about using paper-thin mirrors (maybe on piezoelectric speakers) and as low a frequency as possible to prevent flicker (I would like to produce scrolling script since this seems to be a natural format for curves produced in this manner). I have the electronics worked out for this. I would also like to control the blanking externally.

Another thought that comes to mind is bimetal wire which I have no experience with. I really think that there is a "good" way to do laser writing that is low cost but nobody has thought of it yet. Way-out stereo music (Pink Floyd) applied to the **horizontal** and vertical mirrors assure me that the simple speaker/mirror system is capable of producing the "twists" and "turns" necessary for writing.

--Dale

Msg#: 1287 *GENERAL*

From: STEVE CIARCIA
To: DALE NASSAR
Subj: REPLY TO **MSG# 1204 (LASER WRITING)**

Try using **aluminized** mylar (you can get it in art supply stores). It is shinier than foil, more flexible, and lighter. If you can't find any, let me know.

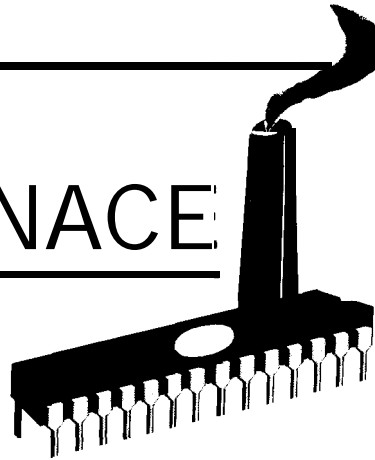
-- Steve

The Circuit Cellar BBS runs on a IO-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, no parity, and either 300, 1200, or 2400 bps.

FIRMWARE FURNACE

Video Signal Timing and Real-Time Interrupt Control

by Ed Nisley



In the world of mainframe computing the term “response time” means the pause you must endure after pressing Enter before you get the results. “Good” response time is less than a second, with “wonderful” falling below half a second. That time is spent fighting through layers of interrupt handlers, communications protocols, and (eventually) the application program itself.

In the world of firmware, though, “response time” is often measured in microseconds. A program responding half a second after an input interrupt may be 499.990 is too late!

This column will explore how the ImageWise receiver produces video synchronizing pulses and updates the video buffer. The stringent timing requires coding tricks that simply aren't taught in structured programming classes so if you're a firm believer in all that, you might want to skip the next few pages.

On the other hand, if you would like to know how to write a real-time interrupt handler that starts

work without saving processor registers, you've come to the right place.

Video Basics

If you're not familiar with TV signals a good discussion of this topic was presented by Steve Ciarcia in the ImageWise project presented in May-June '87 issues of BYTE. Excerpting from that, figure 1 shows video signal timing between the first and second fields of the frame. There are three parts that bear mentioning: the active video, blanking, and the sync pulses. All three are combined into one “composite video” signal that goes to the monitor through the output connector.

Active video is the picture that you see on the screen. The analog voltage represents brightness variations on the screen, with higher voltages being brighter and lower voltages producing darker areas. The voltages are measured with respect to the bottom of the sync pulses, with “whitest white” at about +1.0 volt.

Blanking occurs just before and just after each active video line, as well as during the vertical retrans.

The blanking voltage level is slightly lower than the “blackest black” in the active video signal, so there is no chance an image will appear on the screen. The TV monitor uses blanking times to stabilize the electron beam and position the active video areas.

Finally, sync (short for synchronizing) pulses determine when the monitor will begin horizontal and vertical retraces. The sync pulses are the lowest voltages in the signal and are thus the easiest to detect (a fact used to advantage in the ImageWise transmitter). Horizontal and vertical sync pulses differ only in their timing, not in voltage, so the same circuitry can produce either one by changing the pulse widths.

With those voltages in mind figure 2 shows an expanded view of the horizontal blanking and sync pulse between two active video lines. During the visible part of the picture these pulses occur every 63.5 microseconds, and the whole sequence takes about 12 us from start to finish.

What's not shown on the dia-

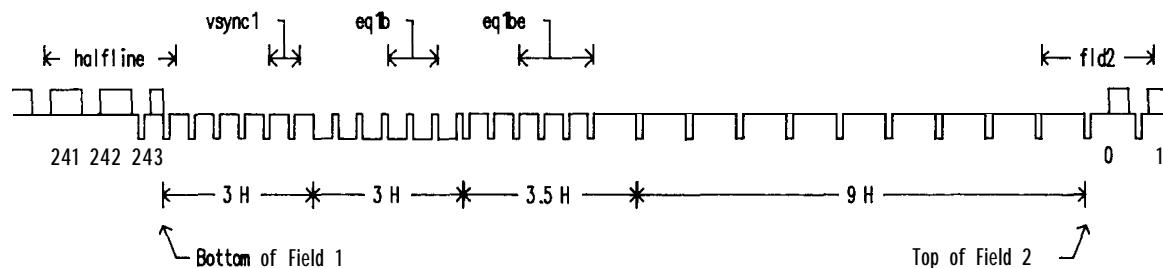


Figure 1 - Video signal timing between the first and second field of a video frame

ram are the sync jitter tolerances. While it's OK to be consistently wrong (by a small amount!), it's not OK to use different timings on each line or in successive fields. Any timing jitter will make the pels quirm on the screen, a condition that's painfully obvious. Therefore, the circuitry and firmware must insure that each sync pulse occurs at absolutely the same time in every frame, with no jitter due to software interference.

Even though the 8031 can execute most instructions in one microsecond, it's obvious that the timings can't be handled by firmware alone. For example, 1.0- μ s instructions can't produce a 63.5- μ s line; especially not by "averaging" a 63-us line with a 64-us one.

Programmable Timing

The ImageWise uses an Intel 8254 Programmable Interval Timer to generate precision sync and blanking pulses. Figure 3 shows the relevant section of the ImageWise schematic.

The 8254 IC has three timers, each with separate inputs and outputs. The inputs start the timers and control how fast the counts occur. Firmware determines the timing modes and durations by setting internal registers for each timer. After the firmware does its work the timers are controlled entirely by the external connections, with the result appearing on the output pin.

Although the video memory is addressed using counters clocked at 5 MHz to produce one pel every 200

nanoseconds, that rate is unreasonably fast for the sync counters. U19 divides the 10-MHz clock by five, giving a 2-MHz clock. That 500-ns clock drives all three 8254 timers, so the minimum timing resolution is half a microsecond and 127 cycles produce a 63.5- μ s video line.

By comparing figures 2 and 3 you can trace the derivation of the timing signals. After the last pel appears in the active video line the address counter overflow signal disables further counting by clearing U12A. The output of U12A drives the video DAC's "blanking" input, so the video signal shifts to the blanking voltage.

A few microseconds after the active video is finished the 8254's Horizontal Sync Period output emits a pulse. That pulse triggers the other two timers, which begin measuring the sync pulse duration and blanking interval.

The Horizontal Sync Duration output sets the video DAC's output to the "sync" level, which is nearly zero volts. The amount of time the Duration output is low controls the sync pulse width, which can be changed in 500-ns increments. It also resets the video address counters to zero in preparation for the next video line.

The Duration output causes an INTO interrupt in the 8031. As we'll see in a moment, the 8031 becomes extremely busy just after that interrupt. But, for the moment, the firmware is just waking up.

The Blanking Duration output determines when the video address counters begin running again by setting U12A. Because Horizontal Sync Duration ends before Blanking Duration occurs, the DAC returns to the "blank" level and the video remains inactive.

After U12A is set, the video buffer begins sending pels to the DAC, which converts them into the appropriate voltages for another scan line. When the counters reach

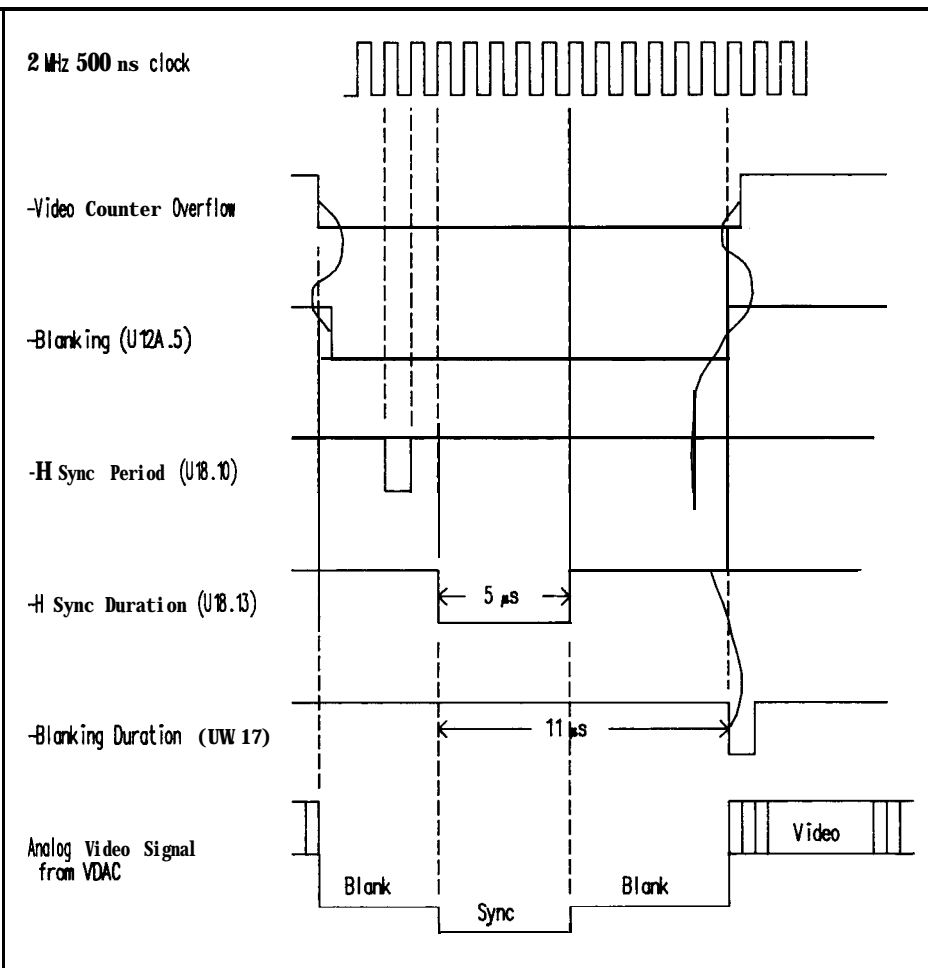


Figure 2 - Horizontal Sync Timing and Blanking Relationship

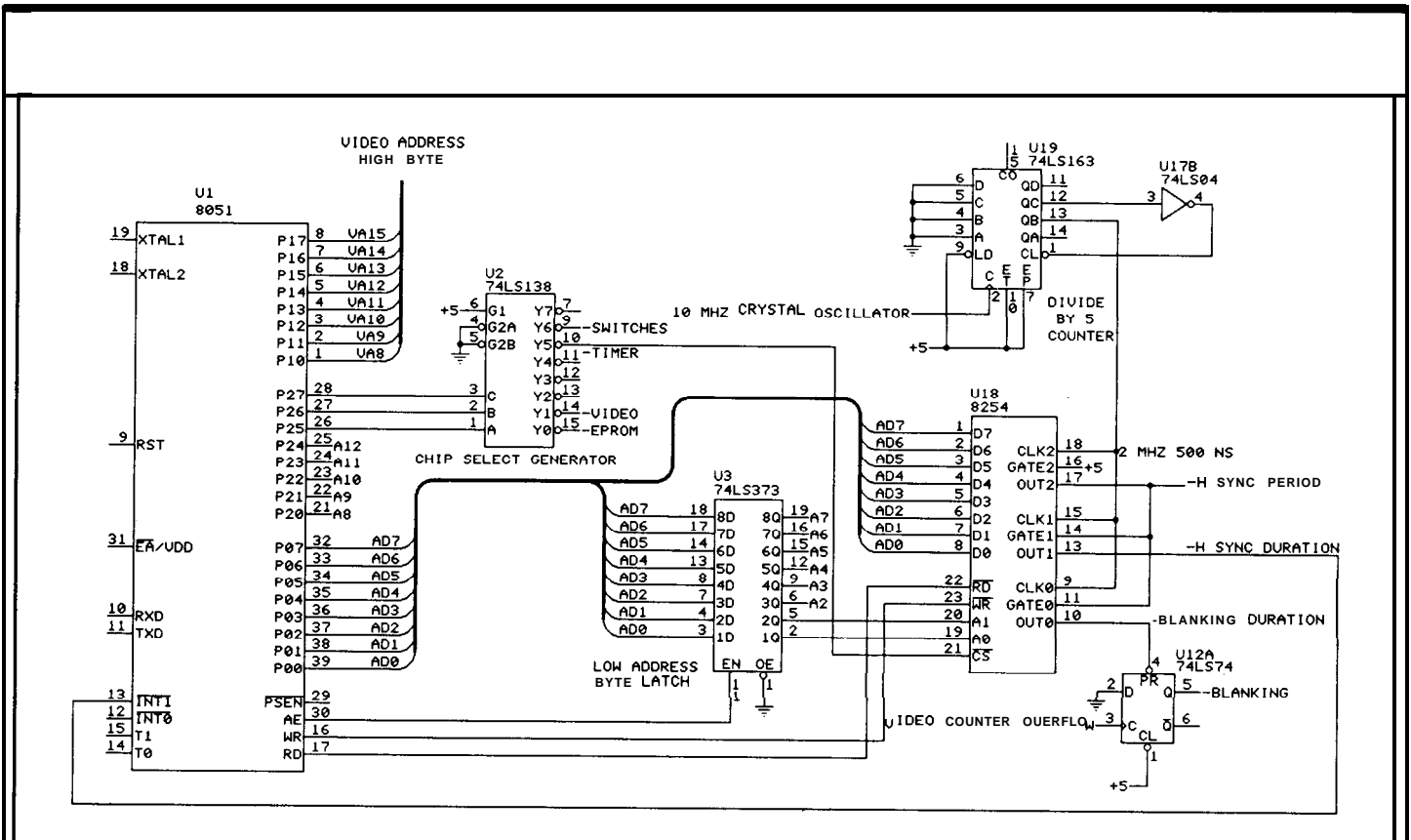


Figure 3 - Section of ImageWise Display/Receiver schematic responsible for generating precision sync timing.

At the end of the line, their overflow will clear U12A and the cycle repeats again.

The firmware is conspicuous by its absence during this discussion. In fact, the 8254 will happily produce perfect line syncs and blanking pulses even if the 8031 is locked up solid! What won't appear, though, are the vertical sync and blanking intervals controlled by the firmware, so the TV image will roll and jump as the monitor interjects its idea of vertical sync into the picture.

Firmware Syncs

The 8031 firmware has two main jobs: set the 8254 registers on the fly to produce vertical timings and write new video data into the RAM buffer when it comes over the serial link. Both of these require extremely close cooperation with the hardware, because both have extremely critical timing requirements.

The ImageWise receiver has a 64K-byte video RAM buffer, divided into 244 lines with 256 pels in each line. The video address counters provide the low-order eight address bits to select the 256 pels in each line, and the 8031 provides the remaining high-order bits to select each line. The 8031 must change its part of the line address at the start of each line before the active video time.

Figure 1 shows what goes on between the two visible fields in each video frame. Line 242 is the last line with a full 63.5-µs duration in the first field. The next line takes only 31.25 µs, and following that are 9.5 line times (9.5 times 63.5 us) that define the vertical sync pulse. The next 9 line times reestablish normal horizontal sync intervals to stabilize the trace, and then the second field begins with a half line of video displayed in a 63.5-µs line time.

The 8031 must change the 8254's register settings whenever the output pulse timings change. In fact, the

changes take effect with the next output pulses, so the firmware must work "one interrupt ahead" to get the timings correct. Each change begins with an INTO interrupt caused by the Horizontal Sync Duration signal from the 8254.

The 8031 supplies both bytes of the RAM address whenever it writes a new pel into the buffer. Multiplexers take the address from either the address counters or the 8031's control signals. Because the RAM can accept only one address, the 8031 must write new data during sync or blanking times to avoid corrupting the image.

INTO causes such a flurry of firmware activity because the 8031 must handle both tasks before the start of active video on each line.

The "main loop" of the ImageWise receiver firmware has only one task: get bytes from the serial interface and write the new data into the video buffer. Because the sync interrupt routine actually writes the

data, the mainline code simply sets up the registers for the write and turns on a flag to indicate that a byte is ready. It waits until the interrupt routine turns the flag off, then sets up the next byte.

When the interrupt occurs, it's a simple matter to test that flag and write the data byte if needed. Listing 1 shows what's involved in the interrupt routine.

Unlike the 8088 and other more advanced processors, the 8031 pushes only the address of the next instruction onto the stack. The interrupt routine is responsible for saving all other registers and flags before changing them. As usual, the penalty for failure to restore a register is program death, usually by dismemberment, when the interrupted routine attempts to continue with corrupted data.

Because there is so little time between the interrupt and the beginning of active video, the firmware takes a novel approach to saving and setting up registers: it doesn't! The 8031 accepts an interrupt on INTO by branching to address 0003H. The hardware can take between 3.3 and 9.8 μ s (at 11.059 MHz) to save the next address and execute the instruction at 0003H. Normally an LJMP instruction would branch to the actual interrupt handler somewhere else in storage, but that LJMP costs two microseconds that can't be spared.

The JBC instruction tests the *takebyte* flag and resets it to zero (if it wasn't on in the first place it's hard to see the change). If *takebyte* was off there is no byte to write and the JBC falls through to the next instruction, which simply branches around the byte write code.

However, if *takebyte* was on, the firmware moves the high byte of the RAM address into the P1 port and writes ACC into External Data Memory using a MOVX. DPH was previously set up so that U2 will select the video RAM buffers and

Listing 1 - Sync Interrupt Routine Part 1

```
**** code from REC.ASM

*--- video sync interrupt code starts here to reduce latency
*   if there's a byte to put in the buffer, do it fast...
*   registers must be set up before takebyte is set:
*       DPH = Video bus address
*       P1data = RAN address high byte
*       DPL = RAM address low byte
*       ACC = data byte value

ORG      0003H                ; INTO handler

JBC      takebyte,dotake     ; test & clear bit
SJMP     notake              ; if off, bypass

dotake   EQU      $           ; was on, stuff byte out
MOV      P1,P1data          ; set up high address
MOVX    @DPTR,A            ; send byte

notake   MOV      P1,P1active ; set new line address

LJMP     syncIRQ1          ; end of time-critical code
```

Listing 2 - Sync Interrupt Routine Part 2

```
**** code from REC.ASM

*-----
* Video timings for 8254 timers
* Units are 0.5 us per tick
* "hperiod" is .25 us too short, so it is adjusted on the fly

nperiod  EQU      127         ; normal sync period
hperiod  EQU      63         ; half sync period

ndur     EQU      10         ; normal sync duration
edur     EQU      5         ; equalizing sync duration
vdur     EQU      54        ; vertical sync duration

nblank   EQU      23        ; normal blanking time
fblank   EQU      255       ; complete blanking time

*-----
* Video sync interrupt handler
* The first part of the handler starts at the vector address 0003
* to eliminate a 2 us LJMP before the time critical code!
* See REC.ASM for that code

syncIRQ1 EQU      $           ; entry from vector code

PUBLIC   syncIRQ1

INC      P1active            ; tick stored address

*--- tick state rep count and decide what to do

DEC      repcount           ; tick counter
XCH      A,repcount         ; is it zero yet?
JNZ      goback             ; not zero, continue

*--- need to handle a state change

XCH      A,repcount         ; restore A
PUSH    ACC                 ; save bystanders
PUSH    PSW
PUSH    DPH
PUSH    DPL

MOV      A,state            ; get state offset
RL      A                   ; get two-byte offset
MOV      DPTR,#stable       ; set up state code base
JMP     @A+DPTR
```

(continued onpage88)

(continued from page 37)

*-- if no state change, return to normal code

```

goback    EQU    $
          XCH    A,repcount    ; restore A and count
          RETI

```

*--- Branch table for sync routines

* State numbers are indexes into this table

```

stable    EQU    $
          ; state numbers
          AJMP   fld1      ; 0 - used only by initialization
          AJMP   fld1      ; 1 - video field 1
          AJMP   halfline  ; 2 - field 1 half line & eql'zing
          AJMP   vsync1    ; 3 - field 1 vertical sync
          AJMP   eqlb      ; 4 - field 1 equalizing
          AJMP   eqlbe     ; 5 - field 1 blanking
          AJMP   fld2      ; 6 - video field 2
          AJMP   eq2a      ; 7 - field 2 equalizing
          AJMP   vsync2    ; 8 - field 2 vertical sync
          AJMP   eq2b      ; 9 - field 2 equalizing
          AJMP   vblank2   ; A - field 2 blanking

```

*-----

* Video sync state routines

**** some code omitted here...

*--- field 1 half line of video

```

* gets control at next-to-last full video line
* does timer writes at sync times to avoid video burps
* merges with the equalizing routine to ensure correct timing
* captures half-line sync to ensure correct timing
• captures first equalizing sync to ensure correct blanking
* half line is -.25us in error

```

```

halfline  EQU    $
half0     JNB    IE0,half0    ; wait for line sync
          CLR    IE0          ; suppress interrupt
          INC    P1           ; tick RAM address

          MOV    DPTR,#I82540
          MOV    A,#hperiod   ; half H sync period
          MOVX   @DPTR,A
          INC    frames       ; tick frame counter

half1     JNB    IE0,half1    ; wait for half-line sync
          CLR    IE0          ; suppress interrupt
          INC    P1           ; tick RAM address
          MOV    Plactive,#maxline+1 ; save RAM data during
          ; blanking

*--- set up for equalizing pulses
* equalizing syncs are -.25us error each

          MOV    DPTR,#I82541
          MOV    A,#edur      ; equalizing sync duration
          MOVX   @DPTR,A
          INC    DPL
          MOV    A,#fblank    ; complete blanking
          MOVX   @DPTR,A

half2     JNB    IE0,half2    ; wait for equalizing sync
          CLR    IE0          ; suppress interrupt

          CLR    blanking     ; force blanking

          MOV    repcount,#4   ; two syncs captured
          INC    state
          LJMP   common

```

(continued on page 39)

multiplex the data out to the RAM. Notice that DPH selects the RAM buffers, but P1 supplies the high byte of the RAM address, which is not the way it's ordinarily done.

Immediately after the MOVX instruction, P1 is set to select the new video line. If this instruction isn't finished by the time active video begins, a "sparkle" will appear on the left edge of the screen.

To put this into perspective, the total elapsed time between the falling edge of the INTO signal and the MOV instruction setting a new value into P1 can be 13 to 19 μ s. You can see that there will almost always be a little sparkle because there are only 11 μ s allowed. Many monitors hide the sparkle in the overscan area behind the bezel, but it's easy to spot on a high-quality underscanned monitor. It occurs only when new data is written into the buffer, so there's no sparkle during normal viewing.

You're probably wondering why we didn't use the Horizontal Sync Period signal to trigger the interrupt. That signal would give the firmware another 1.5 μ s after the interrupt, which is a significant fraction of the total time. It turns out that the INTO signal must be active for at least one machine cycle (1.09 μ s at 11.059 MHz) to ensure that the hardware will act on it, and Horizontal Sync Period is active for only 250 ns.

Similarly, the Video Counter Overflow signal can't be used because it occurs only on lines with active video, not on the completely blanked lines during vertical retrace. That's what tradeoffs are all about!

The LJMP to the rest of the interrupt handler takes place after P1 is set up. There are other interrupts vectored into the addresses near 0000H, and it was a good thing that the time-critical INTO code didn't overlap any of those addresses!

Listing 3 shows the section of code that sets up all the registers for the interrupt routine. The *takebyte* flag serves to alert the interrupt code that there's a byte ready to write, and also tells the mainline code when the byte's been written. The interrupt routine increments the RAM data address value so that the mainline routine can simply reset the flag to write duplicate values.

Marking Time

Fortunately, it gets easier from here because the timings are less critical. Listing 2 shows some of the remainder of the interrupt handler code that runs after that in listing 1.

The first order of business is to increment *Pactive*, which holds the high-order byte of the next video line's address. Doing this first ensures that it's ready for the next sync interrupt no matter what else goes on. *Pactive* gets reset to zero at the bottom of each field by other code.

Most video lines don't require a timer change, so the code simply decrements *repcount*, which counts the number of sync interrupts before the next change. The zero test in JNZ examines the entire accumulator, rather than a zero flag set as in many other micros, so the XCH swaps the current contents of ACC with *repcount* before the JNZ. If it's not zero yet, the code simply returns to the interrupted mainline code using a RETI after restoring ACC.

When *repcount* hits zero, the firmware starts looking more like a normal routine: it saves the registers (at last!) and uses a variable called *state* to decide what timer changes are required. *State* is an index into a table of branches to subroutines: the JMP @A+PC adds *state* (in the ACC) to the current instruction pointer and begins executing at that address.

Figure 1 shows the routines that handle various parts of the sync

(continued from page 38)

```

*--- field 1 vertical sync
* gets control after 5th equalizing sync
* updates internal copy of DIP switch settings
* captures 6th sync to ensure correct timing
* vertical syncs are +.25us error
vsyncl      EQU      $
            MOV      Plactive,#maxline+1 ; save RAM data during
            ; blanking
            MOV      DPTR,#switches ; pick up switch state
            MOVX     A,@DPTR
            CPL      A ; flip bits to get ON = 1
            MOV      swcopy,A ; update RAM copy
vs1         JNB      IE0,vs1 ; wait for 6th equalizing
            ; sync
            CLR      IEO ; suppress interrupt
            MOV      DPTR,#I82540
            MOV      A,#(hperiod+1) ; half-horizontal period
            INC      DPL
            MOV      A,#vdur ; vertical sync pulse
            ; duration
            MOVX     @DPTR,A
            MOV      repcount,#4 ; one sync caught
            INC      state
            LJMP     common
*--- field 1 second equalizing pulse, short set
* gets control after 4th vertical sync
* captures 5th sync
* sets 6th v sync period to -.25us error
* captures 6th sync to ensure correct timing
* the first three equalizing pulses are -.25us error
eq1b        EQU      $
            MOV      Plactive,#maxline+1 ; save RAM data during
            ; blanking
eq1b1       JNB      IE0,eq1b1 ; wait for 5th vertical sync
            CLR      IEO ; suppress interrupt
            MOV      DPTR,#I82540
            MOV      A,#hperiod ; half horizontal sync
            MOVX     @DPTR,A
eq1b2       JNB      IE0,eq1b2 ; wait for 6th vertical sync
            CLR      IEO ; suppress interrupt
            MOV      DPTR,#I82541
            MOV      A,#edur ; equalizing sync duration
            MOVX     @DPTR,A
            MOV      repcount,#3 ; two syncs caught
            INC      state
            LJMP     common
*-transition to full horizontal line timing
* gets control after 3rd equalizing sync
* captures 4th sync, changes equalizing sync period to +.25us error
* captures 5th sync, changes sync period
* captures 6th sync, merge with vertical blanking setup
eq1be        EQU      $
            MOV      Plactive,#maxline+1 ; save RAM data during
            ; blanking
            MOV      DPTR,#I82540
            MOV      A,#(hperiod+1) ; half horizontal period
            MOVX     @DPTR,A
eq1bel1     JNB      IE0,eq1bel1 ; wait for 4th sync
            CLR      IEO ; suppress interrupt
eq1be2      JNB      IE0,eq1be2 ; wait for 5th sync
            CLR      IEO ; suppress interrupt
    
```

(continued on page 40)

(continued from page 89)

```

MOV DPTR,#I82540
MOV A,#nperiod ; normal H sync period
MOVX @DPTR,A

eqlbe3 JNB IEO,eqlbe3 ; wait for 6th sync
CLR IEO ; suppress interrupt

*--- field 1 vertical blanking

MOV DPTR,#I82541
MOV A,#ndur ; normal sync duration
MOVX @DPTR,A

MOV repcount,#9 ; no syncs pending
INC state
LJMP common

*--- top of active video for field 2
* capture next sync here to ensure correct unblinking

fld2 EQU $
MOV P1,#maxline+1 ; save RAM data

MOV DPTR,#I82542
MOV A,#nblank ; normal blanking duration
MOVX @DPTR,A

MOV P1,#00H ; preset the address

fld2a JNB IEO,fld2a ; wait for sync
CLR IEO ; suppress interrupt

*--- field 2 half line of video
* blanking is released after about half a line
* capture next sync to allow enough time for return

fld2b1 MOV A,#10 ; delay to about mid line
DJNZ ACC,fld2b1

SETB blanking ; release blanking

fld2b2 JNB IEO,fld2b2 ; wait for next sync
CLR IEO ; suppress interrupt
INC P1 ; tick RAM address

MOV Plactive,#2 ; for next interrupt
MOV repcount,#maxline-2 ; two syncs captured
INC state
LJMP common

```

signal. By comparing the code in listing 2 with the timings in figure 1 you should be able to figure out what's going on.

If you're following closely, you'll wonder how to create a precise half-line sync from a 500 ns clock. After all, half of 63.5 is 31.75. Where does the extra 250 ns come from?

It turns out that monitors can stand a little jitter during vertical retrace and it's all invisible because the beam is blanked. The firmware makes some lines 250 ns too long and an equal number 250 ns too short, so the average is correct in each field. Both vertical retraces include about nine completely blanked lines before the active video lines, which is enough time to stabilize the circuitry.

Why not use a 4-MHz clock to get a 250-ns period, which would make the half-lines come out exactly right? That would require another 74LS74 flipflop, and hardware is expensive!

Many of the routines during vertical retrace capture interrupts rather than execute a RETI. There are only 31.5 µs between interrupts, which translates to perhaps two dozen instructions at 11.059 MHz. There's simply not enough time to go though all of the normal processing shown in figures 1 and 2 in addition to changing the registers. So, for short bursts, the video interrupt simply spins on the sync interrupt input.

This has an interesting side effect: because the serial interrupt has a lower priority than the video sync interrupt, it is locked out during those spin loops. If the code spins too long, a serial input overrun will occur and an incoming byte will vanish.

Quirks

When you study the listings for a while you'll find some other inter-

Listing 3 - Write new byte into buffer

```

**** code from VIDEO.ASM
*--- present data bytes to the video interrupt handler
* A contains the video data byte
* B contains the rep count for the byte
* DPTR is set up for the video interrupt:
* DPH = video bus address
* DPL = pel address in line
* Pldata contains the high byte of the video buffer address
* DPTR is returned pointing to the next pel, wraps to 00 at end

dodata EQU $
RL A ; move video to high 6 bits
RL A
ANL A,#0FCH ; strip low bits

redo EQU $
SETB takebyte ; interlock with video interrupt

takewait J B takebyte,takewait ; ... wait for completion
INC DPL ; tick address within line only

DJNZ B,redo ; tick count and repeat
RET

```


esting quirks and peculiarities. For example...

Because the RAM Write line isn't gated, any write to the timers will also write whatever's on the RAM data lines into whatever location is called for by the RAM address pins. As you might expect, this can result in random data corruption. The solution is simple: the timer writes occur during the blanked lines, with the RAM address set to an unused line. The sole exception, on the last half-line in the first field, trashes only the first byte in that line because it's written immediately after the interrupt before the active video starts.

The first active video line in Field 2 is 63.5 μ s long, but the first half is blanked. The firmware handles this timing with a software loop, which gives acceptable accuracy because the sync pulses aren't affected. You can see the resulting jitter by adjusting your monitor's vertical hold control to roll the image down.

All in all, though, the ImageWise sync generation produces a rock solid picture with a minimum amount of specialized hardware. In later columns, I'll describe how the serial interrupt handler fits into the sync generator without trashing anything. For starters, imagine what happens if a serial interrupt occurs just after the *takebyte* flag goes on, just before the video interrupt handler gets control... ■

Editor's Note: Steve Garcia often refers 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, (203) 875-2751

Garcia'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.

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

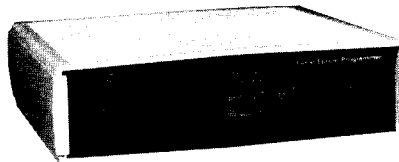


unretouched photos

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\$49.00

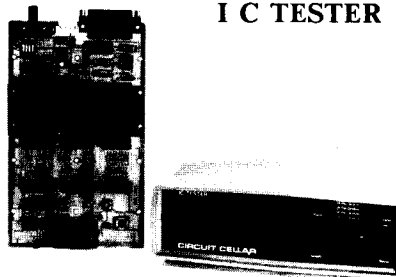
SERIAL EPROM PROGRAMMER



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

I C 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

Circuit Cellar Inc-4 Park St., Suite 12-Vernon, CT 06066 (203) 875-2751

SOFTUART

Software Emulation of Full-Duplex Serial Channels

by Bill Curlew

I'm sure you've heard the expression "like trying to put ten pounds in a five-pound bag." Such a statement is appropriate for some materials, but not all. In the field of digital electronics, when we "fill" the "bag" with more performance and capability, we usually have to pay the price with more board space and hardware, and the added features never fit in the same "bag."

At certain stages in the development of a product, the increasing complexity caused by redefining tasks and capabilities can be managed and dealt with. Unfortunately, here comes a time when board space is restricted and adding new functions in hardware is not possible.

Such was the case that Steve Ciarcia and I had when we were working on a project together a couple years ago. I don't remember exactly why we were building this particular device but it was the process of getting to the end that is relevant here.

Like many of his other projects, this one used an 8031 microcontroller with external memory and I/O. The circuit connected between a modem and a computer and performed a little preprocessing on the data as it passed through the device. In any case, such a connection required two full-duplex serial channels.

Initially, we investigated adding hardware for the two serial ports (or adding hardware and using the 8031's on-chip serial port as one of

them). Worst case, this meant designing in either two additional 40-pin UARTs (Universal Asynchronous Receiver/Transmitter) or a single 40-pin SIO (Serial Input/Output) that contains the equivalent of two UARTs. Other necessary support chips would have included a few 16-pin chips for address decoding, and a clock/counter chip with a crystal to provide the timing for the serial chips.

The price of the added hardware was prohibitive and appeared to take up too much room. Steve reminded me that I once did a software emulation of a serial port on a Z8 and suggested doing the same for the 8031.

I love a challenge. I have always been fascinated by the UART and its ability to do complex functions in hardware and this looked like the perfect opportunity to put my practical knowledge of the UART device to the test. Ultimately, we built the device, complete with serial ports emulated in software, and everything worked perfectly.

I also believe in well-documented code. While working on this project I wrote up a set of general-purpose routines for emulating two UARTs. Steve convinced me that this documentation would make a good article. While specifically coded for an 8031 processor, the logic used to emulate these UARTs (actually a combination of different UARTs that I have worked with over the years) should serve as an educational tool as well as a guide for applying

this same technique elsewhere.

What is a UART?

A UART is used to transmit and receive data serially between computers, terminals, modems, and so on. UARTs provide the following functions:

- Parallel-to-serial / serial-to-parallel conversion of data
- Addition and deletion of framing, i.e., the start and stop bits surrounding the data. Framing bits ensure that the beginning and end of the byte have been detected correctly.
- The generation of a *parity* bit, which is used to provide a rudimentary validation of the data. Parity in serial communications works exactly the same way as the parity bits in the memory of many computers.
- Buffering of transmitted and received data (usually only one byte in each direction). This gives the computer more time to load new bytes to be transmitted into the UART, and to pick up received bytes before they are overwritten by the next incoming byte.
- Providing hardware handshaking by the use of control signals like TBMT (Transmit Buffer Empty), TXRDY (Transmitter Ready), and DAV (Data Available). These will be discussed below.
- Providing notification of errors in framing (FE), parity (PE) and overruns (OVR). These, too, will be discussed later.

- Not usually present in a UART, but available in an SIO, is the ability to manipulate and detect modem control signals like RTS/CTS (Ready To Send/Clear To Send), DTR/DSR (Data Terminal Ready/Data Set Ready), and DCD (Data Carrier Detect). My software allows you to set RTS and read CTS, but the operation of the UART itself is independent of the state of these signals.

UART options

UART parameters such as data transfer rate, number of data bits, parity, and so on can usually be configured from software. Matching these parameters to those being used by the remote device can often be the most difficult part of establishing a connection. Nowadays, this has been made simpler by autodetection routines that automatically determine the data transfer rate and parity, then reset the local UART or SIO to conform with the incoming format. It is this automatic configuration software

that requires you to send a known character like a space or a carriage return when first logging on.

Here are the options that I have made available on this software encoded UART:

- Set number of data bits being sent and received to 5, 6, 7, or 8
- Enable or disable parity bit transmission and reception
- If parity is used, optionally generate and test for odd, even, mark (always one), or space (always 0) parity.
- Send 1 or 2 stop bits.

UART Operation

The UART is driven by a clock that divides up each bit into several slices. In real UART and SIO devices the frequency of this clock will typically be 8 or 16 times the data transfer rate. UART transmission involves sending out the framing, data, and parity bits at the start of each bit time. Receiving involves detecting the start of a byte by waiting for a start bit, then shifting

in the other bits of the byte and doing the parity and framing checks. While waiting for the start bit, if the UART detects a high-to-low transition (normally indicating the beginning of the start bit), but it turns out that it was really not a start bit, the UART will return to hunting for a start bit. This is called false start bit detection.

Figures 1 and 2 are state diagrams that represent a UART's operation. The diagrams show what operational states the UART goes through to perform serial input and output. The lines and arrows show the events that drive the UART code through the various states. Initialization of the UART forces it to the IDLE state for both transmit and receive.

UART State Control

As mentioned above, a UART divides each data bit into smaller bit slices. Since the UART states described below and illustrated by the state diagrams deal at the bit level, there is a control routine that checks

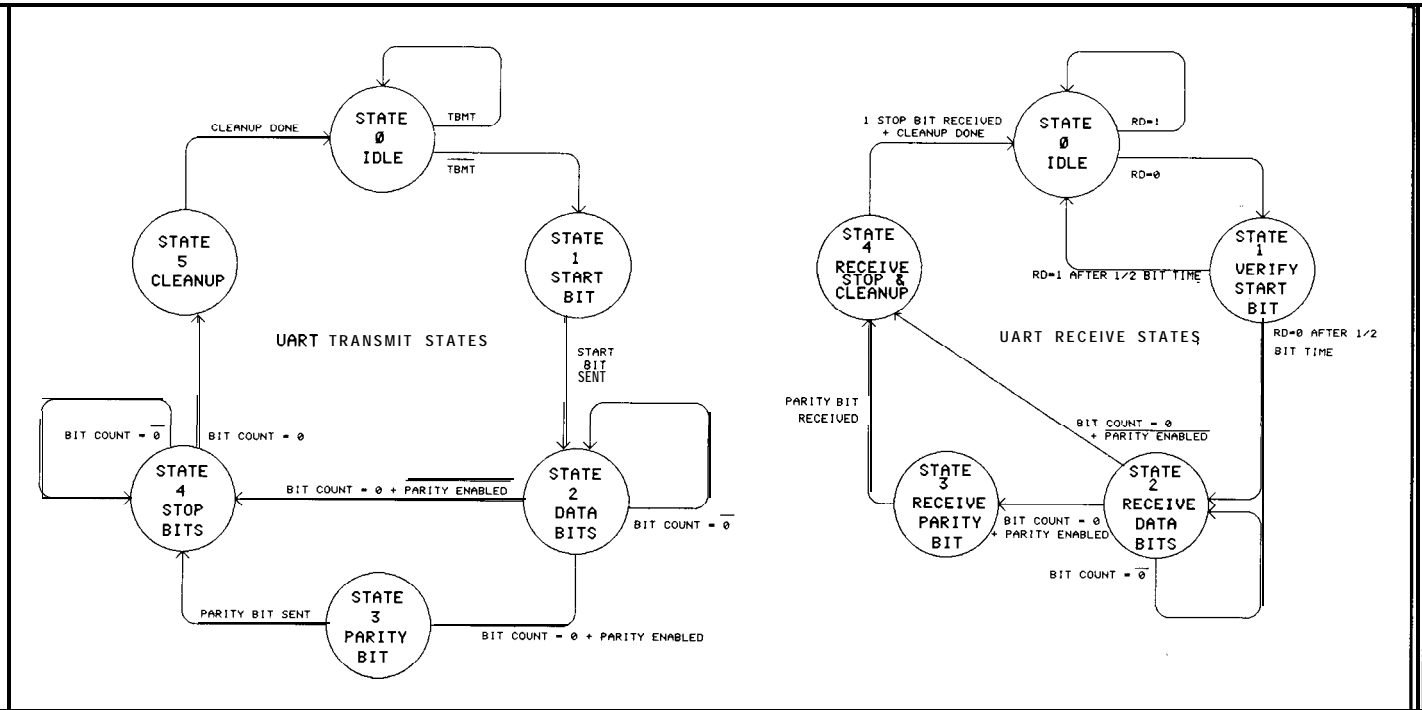


Figure 1 and 2 - State Diagrams for UART transmit and receive functions

at each bit slice to determine if a transition from one UART state to another should take place. This control routine is interrupt driven by a counter/timer in the 8031. The operation of this interrupt routine is shown in flowchart 1, and the actual code to do this is illustrated in code segment 1. Since there are really two UARTs running at the same time, the interrupt routine checks both logical devices. Note that the transmit states for both UARTs are given priority over the receive states. This is to ensure that the edges of the transmitted bits have as little jitter as possible. Jitter is caused when varying amounts of code are executed between the reception of the interrupt and the change in the transmitted bit level.

UART Transmit States

State 0, Idle

In the idle state, the UART checks its input buffer at each bit slice time. When data is detected in the buffer, (indicated by the reset of the TBM signal), the data is moved to the transmitter register. The UART then transfers to state 1, initiating a start bit.

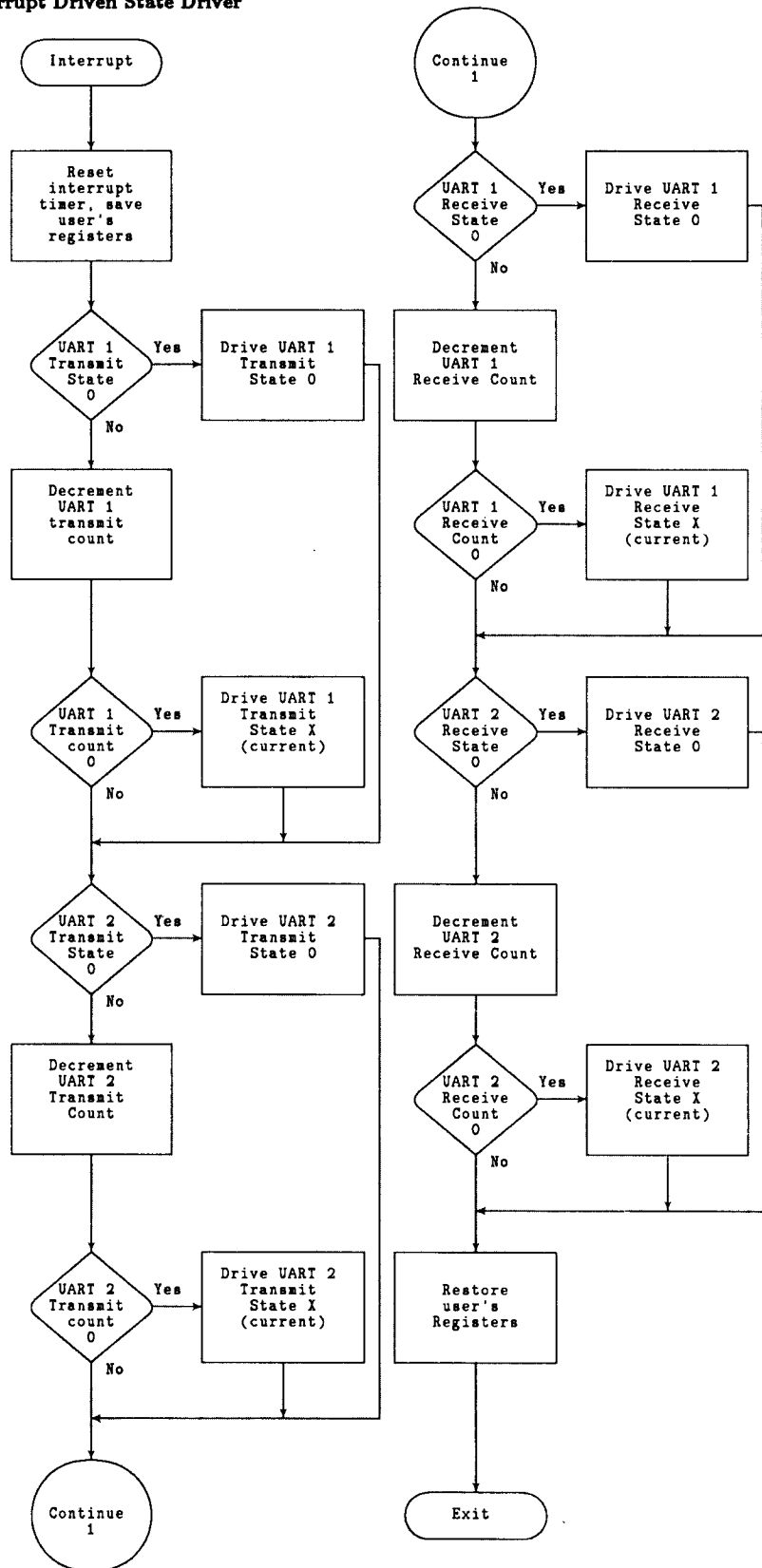
State 1, Start Bit

The UART begins a start bit by setting the output line to a logic 0. A counter is loaded with the number of bits to be sent (in this case, one), and another counter is set to the number of bit slices needed to transmit a whole bit. When the bit has been transmitted, the UART transfers to state 2, Send Data Bits.

State 2, Data Bits (see flowchart 2)

The UART knows the number of data bits that need to be transmitted since this was set during initialization. As each bit is transmitted, the data bit count is decremented. When all data bits have been sent, the UART determines if a parity bit

Flowchart #1
Interrupt Driven State Driver



has been requested. If parity has not been requested, the UART transfers to state 4. Otherwise, state 3 is entered.

State 3, Parity Bit

The UART determines the type of parity to be generated and sets the appropriate bit level for that parity. When the parity bit has been transmitted, the UART enters state 4.

State 4, Stop Bit(s)

Either one or two stop bits can be sent, as specified in the initialization options. When all stop bits have been sent, the UART enters state 5.

State 5, Cleanup

At the end of character transmission, the UART goes through cleanup processing. This involves setting TXRDY to 1, clearing the transmit bit counts and the count of bit slices remaining, and setting up for the transfer to state 0.

UART Receive States

State 0, Idle

During the receive idle state, the status of the incoming serial line is monitored. State 0 continues for as long as a logic-1 level is being received. When the serial input make the transition to a logic 0, the UART sets up a count of bit slices that make up one half of a bit time. After one half of a bit time has passed, state 1 is entered.

State 1, Validate Start Bit

When one half of a bit time has passed, the UART again samples the incoming serial line. If a logic 1 is read, it is assumed that this is not a valid start bit and the UART goes back to the idle state. If the UART receives a logic 0, then it is assumed that this is a valid start bit and the UART will delay for a time equivalent to a full bit. This delay causes the UART to sample the remaining

Code Segment 1

```

; UART INTERRUPT HANDLING ROUTINE
;
;
; ***** GOT TO PUSH YOUR OWN PSW WHEN IN 8031 *****
;
;     PUSH PSW           ; SAVE OLD PSW
;     PUSH ACC           ;
;
; SET UP THO COUNTS BASED ON BIT RATE
;
;     MOV THO,TRELOAD    ; PUT COUNTS IN THO
;     PUSH B             ;
;     PUSH DPL           ;
;     PUSH DPH           ;
;
;
; CHECK UART STATES FOR BOTH UARTS, AND ENTER THE STATE
; REQUIRED, DO THE TRANSMIT SIDES OF BOTH UARTS FIRST TO
; MINIMIZE JITTER ON THE OUTPUTTED DATA.
; (RECEIVING IS MORE TOLERANT THAN TRANSMITTING)
;
TUART1 EQU $
MOV A,TSTAT1 ; PICK UP TRANSMIT STATE1
CJNE A,#0,U1TCOUNT ; IF STATE NOT 0, GO ON
CALL U1TSTAT0 ; DO STATE 0
S JMP TUART2 ; DOUART 2
U1TCOUNT EQU $
MOV A,TCOUNT1 ; GET COUNTS
DEC A ; COUNT=COUNT-1
MOV TCOUNT1,A ; OR IN NEW COUNTS
CJNE A,#0,TUART2 ; IF COUNT 00 THEN DO UART 2
STATET1X EQU $
MOV A,TSTAT1 ; PICK UP TRANSMIT STATES
MOV DPTR,#U1TTABLE ; GET TABLE ADDRESS
MOVC A,@A+DPTR ; GET JMP OFFSET
JMP @A+DPTR
U1TTABLE EQU $
DB 00H ; NEVER INDEX OF ZERO
DB U1TTEST1-U1TTABLE
DB U1TTEST2-U1TTABLE
DB U1TTEST3-U1TTABLE
DB U1TTEST4-U1TTABLE
DB U1TTEST5-U1TTABLE
;
U1TTEST1 EQU $
CALL U1TSTAT1 ; DO STATE 1
S JMP TUART2 ; DO UART 2
U1TTEST2 EQU $
CALL U1TSTAT2 ; DO STATE 2
S JMP TUART2 ; DOUART 2
U1TTEST3 EQU $
CALL U1TSTAT3 ; DO STATE 3
S JMP TUART2 ; DOUART 2
U1TTEST4 EQU $
CALL U1TSTAT4 ; DO STATE 4
S JMP TUART2 ; DOUART 2
U1TTEST5 EQU $
CALL U1TSTAT5 ; DO STATE 5
TUART2 EQU $
MOV A,TSTAT2 ; PICK UP TRANSMIT STATES
CJNE A,#0,U2TCOUNT ; IF STATE NOT 0, GO ON
CALL U2TSTAT0 ; DO STATE 0
S JMP RUART1 ; DO RECEIVE UART 1
U2TCOUNT EQU $
MOV A,TCOUNT2 ; GET COUNTS
DEC A ; COUNT=COUNT-1
MOV TCOUNT2,A ; OR IN NEW COUNTS
CJNE A,#0,RUART1 ; IF COUNT 00 THEN DO RECEIVE UART 1
STATET2X EQU $
MOV A,TSTAT2 ; PICK UP TRANSMIT STATES
MOV DPTR,#U2TTABLE ; GET TABLE ADDRESS
MOVC A,@A+DPTR ; GET JMP OFFSET
JMP @A+DPTR

```

(continued on page 46)

(continued from page 45)

```

U2TTABLE EQU $
    DB OOH ; NEVER INDEX OF ZERO
    DB U2TTEST1-U2TTABLE
    DB U2TTEST2-U2TTABLE
    DB U2TTEST3-U2TTABLE
    DB U2TTEST4-U2TTABLE
    DB U2TTEST5-U2TTABLE
;
U2TTEST1 EQU $
    CALL U2TSTAT1 ; DO STATE 1
    SJMP RUART1 ; DO RECEIVE UART 1
U2TTEST2 EQU $
    CALL U2TSTAT2 ; DO STATE 2
    SJMP RUART1 ; DO RECEIVE UART 1
U2TTEST3 EQU $
    CALL U2TSTAT3 ; DO STATE 3
    SJMP RUART1 ; DO RECEIVE UART 1
U2TTEST4 EQU $
    CALL U2TSTAT4 ; DO STATE 4
    SJMP RUART1 ; DO RECEIVE UART 1
U2TTEST5 EQU $
    CALL U2TSTAT5 ; DO STATE 5
; NOW DO RECEIVE STATES FOR BOTH UARTS
RUART1 EQU $
    MOV A,RSTAT1 ; PICK UP RECEIVE STATES
    CJNE A,#0,U1RCOUNT ; IF STATE NOT 0, GO ON
    CALL U1RSTAT0 ; DO STATE 0
    SJMP RUART2 ; DO UART 2
U1RCOUNT EQU $
    MOV A,RCOUNT1 ; GET COUNTS
    DEC A ; COUNT=COUNT-1
    MOV RCOUNT1,A ; OR IN NEW COUNTS
    CJNE A,#0,RUART2 ; IF COUNT 00 THEN DO UART 2
STATER1X EQU $
    MOV A,RSTAT1 ; PICK UP RECEIVE STATES
    MOV DPTR,#U1RTABLE ; GET TABLE ADDRESS
    MOVC A,@A+DPTR ; GET JMP OFFSET
    JMP @A+DPTR
U1RTABLE EQU $
    DB OOH ; NEVER INDEX OF ZERO
    DB U1RTEST1-U1RTABLE
    DB U1RTEST2-U1RTABLE
    DB U1RTEST3-U1RTABLE
    DB U1RTEST4-U1RTABLE
U1RTEST1 EQU $
    CALL U1RSTAT1 ; DO STATE 1
    SJMP RUART2 ; DO UART 2
U1RTEST2 EQU $
    CALL U1RSTAT2 ; DO STATE 2
    SJMP RUART2 ; DO UART 2
U1RTEST3 EQU $
    CALL U1RSTAT3 ; DO STATE 3
    SJMP RUART2 ; DO UART 2
U1RTEST4 EQU $
    CALL U1RSTAT4 ; DO STATE 4
RUART2 EQU $
    MOV A,RSTAT2 ; PICK UP RECEIVE STATES
    CJNE A,#0,U2RCOUNT ; IF STATE NOT 0, GO ON
    CALL U2RSTAT0 ; DO STATE 0
    SJMP INTEXTIT ; DO EXIT
U2RCOUNT EQU $
    MOV A,RCOUNT2 ; GET COUNTS
    DEC A ; COUNT=COUNT-1
    MOV RCOUNT2,A ; SAVE NEW COUNT
    CJNE A,#0,INTEXTIT ; IF COUNT 00 THEN DO EXIT
STATER2X EQU $
    MOV A,RSTAT2 ; PICK UP RECEIVE STATES
    MOV DPTR,#U2RTABLE ; GET TABLE ADDRESS
    MOVC A,@A+DPTR ; GET JMP OFFSET

```

(continued on page 47)

bits in the middle of each bit. After the one-bit delay is set up, the UART goes to state 2.

State 2, Receive data bits (see flowchart 3)

The number of data bits to be received has been set as part of UART initialization. The UART samples the serial input line once at each bit interval and shifts the bit value received into a receive buffer. Then the count of bits to be received is decremented. When all bits have been received, the UART checks to see if parity is expected. If parity is not expected, the PE flag is cleared and the UART transfers to state 4. If parity is expected, the UART transfers to state 3.

State 3, Receive and Check Parity

The UART samples the serial input and stores the received parity bit. The parity of the previously received data bits is then generated for comparison with the bit just read. The type of parity expected is set up during initialization, and the received parity is matched against what was expected. The PE flag is set or reset depending on the match between expected and received parity. The UART now enters state 4.

State 4, Stop Bits and Cleanup (see flowchart 4)

In this state, the UART samples the serial input line, expecting to see a stop bit (logic 1). This test is only done once, even if two stop bits are expected. I set it up this way because it is how most hardware UARTs work. The reasoning is that the computer gets notified of available data one bit-time early, which gives it more time to process between received characters. This helps in preventing overrun situations.

When the stop bit has been read, FE is set or reset depending on whether or not the stop bit received

(continued from page 46)

```

        JMP @A+DPTR
U2RTABLE EQU $
        DB OOH                ; NEVER INDEX OF ZERO
        DB U2RTEST1-U2RTABLE
        DB U2RTEST2-U2RTABLE
        DB U2RTEST3-U2RTABLE
        DB U2RTEST4-U2RTABLE
;
U2RTEST1 EQU $
        CALL U2RSTAT1        ; DO STATE 1
        SJMP INTEXTIT        ; GOT0 EXIT
U2RTEST2 EQU $
        CALL U2RSTAT2        ; DO STATE 2
        SJMP INTEXTIT        ; GOT0 EXIT
U2RTEST3 EQU $
        CALL U2RSTAT3        ; DO STATE 3
        SJMP INTEXTIT        ; GOT0 EXIT
U2RTEST4 EQU $
        CALL U2RSTAT4        ; DO STATE 4
;
; ALL UART STUFF DONE, RESTORE REGS AND EXIT

; POP SAVED REGISTERS FROM THE STACK
;
INTEXTIT EQU $
        POP DPH                ;
        POP DPL                ;
        POP B                   ;
        POP ACC                 ;
        POP PSW                 ;

        RET1                    ; RETURN FROM INTERRUPT

,      END OF UINVEC ROUTINE

```

was a logic 0 or 1. The UART now checks DAV to see if the previously received byte is still in the buffer. If DAV is true, (the last byte is still in the buffer), the OVR flag is set. Otherwise, OVR is reset. If fewer than eight data bits were received, the data byte will be right justified and padded on the left with binary 0s. DAV is set true and the byte is moved to the receive buffer. Cleanup then occurs and the UART returns to state 0.

Transmitting and receiving data with a UART

To transmit and receive data using a UART you must first initialize the UART to set the options described above. On a real device, it is good idea to read one byte from the UART data port and throw it away to reset any leftover error indicators at this point.

Transmitting involves the following steps:

- 1) Set the RTS (Request To Send) signal to logic 1
- 2) Read the UART status register and test the CTS (Clear To Send) signal
- 3) If CTS is false, go to (2)
- 4) Read the UART status port and test the TBMT control signal
- 5) If TBMT is false, go to (4)
- 6) Send data to the UART data port

The RTS and CTS signals are necessary only if the attached hardware requires them. In most cases, the modem or other device can be configured to ignore these signals.

Transmit Buffer Empty (TBMT), is true whenever the UART is ready to buffer another byte of data to send. Another signal that could be used is Transmitter Ready (TXRDY). TXRDY indicates that no data is currently being shifted out through the transmitter. Generally, TBMT is used since it allows the UART to keep one byte buffered on the side while another is being shifted out of

the transmitter. This keeps the data flowing out of the UART at the fastest possible rate.

Receiving data is a bit more complicated due to the error conditions involved. Receiving involves the following steps:

- 1) Read the UART status register and test for the error conditions FE, PE, and OVR.

- 2) If any of these conditions exist, read the data port and return the data and error indicator(s) to the main program. (The main program might elect to ignore the error, attempt recovery, or alert the user of the system.)

- 3) If DAV is not set, go to (1)

- 4) Read the data and return it to the main program.

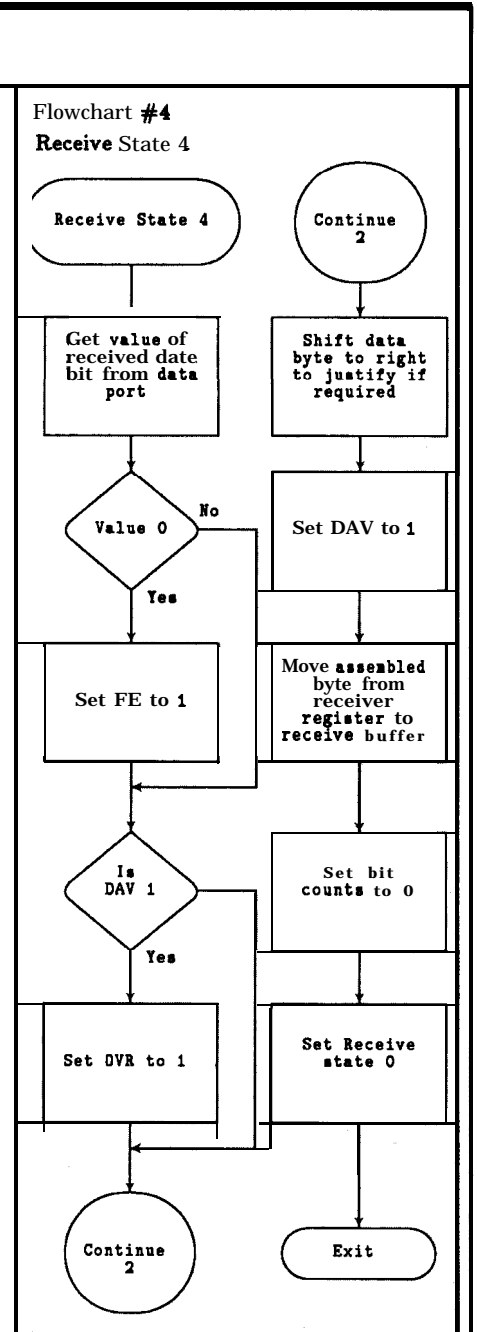
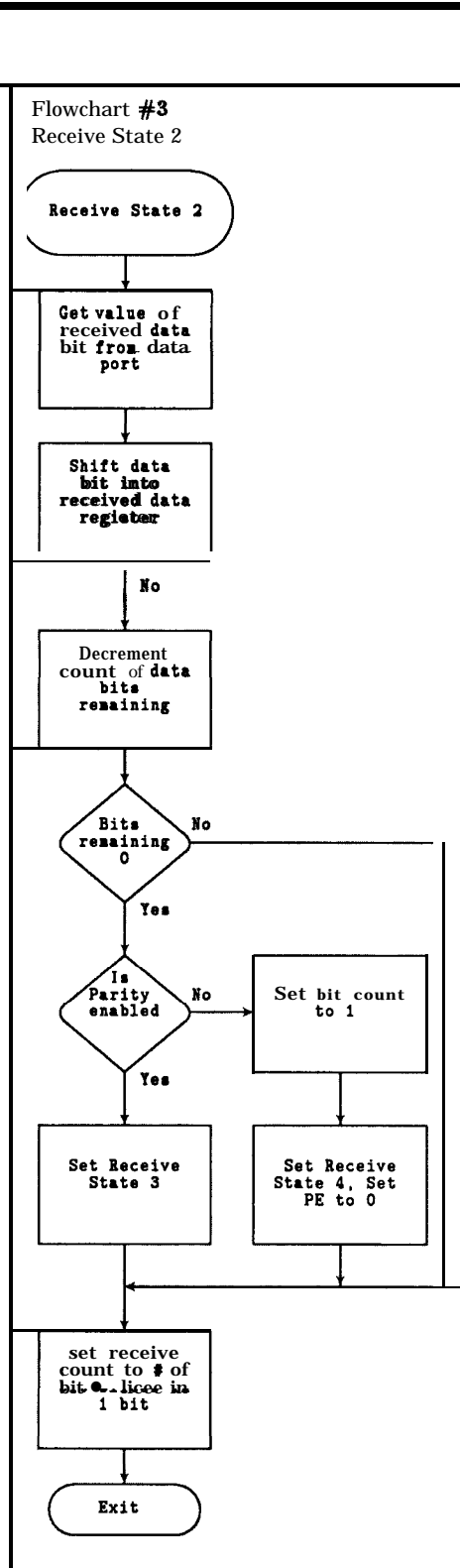
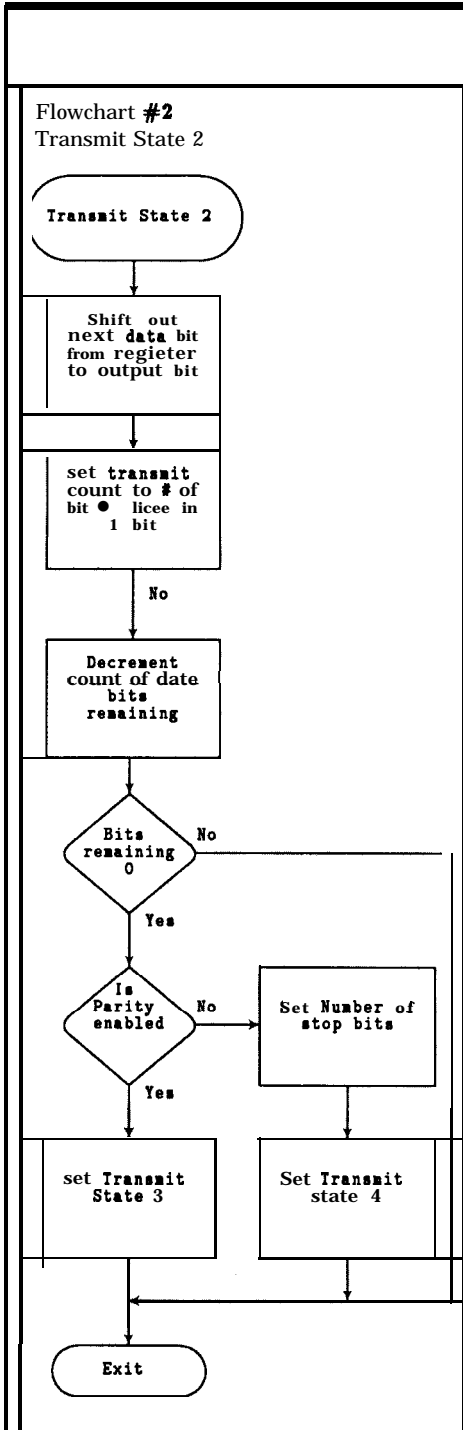
The error indicators each represent a particular problem the UART has detected. FE is a Framing Error. This is caused when the UART expects to receive a stop bit (logic 1) and instead gets a logic 0. This means the data shifted into the receiver did not start on a byte boundary and is no good.

PE stands for Parity Error. This means that the parity bit did not match what was expected according to the options set in the UART. Your software might elect to ignore this condition if parity settings are not critical and no framing error occurred.

OVR stands for Overrun. An OVR error occurs when the computer doesn't read the data from the UART before the next byte being received had to be moved to the receiver register. The previous contents have been wiped out and the old data byte has been lost.

Special Considerations

The software that I have written emulates two UART devices at once, thus providing the dual input and output channels required for



Steve's project. It was written specifically for the 8031 but is also applicable to the 8052. Register usage would have to change to allow concurrent use with BASIC-52, however.

The clock source must be stable and should be a hardware or timer interrupt if possible. On a processor like the Z8, the clocks can be set up to automatically reload their timer values at each interrupt, so can give

a timebase that is totally independent of software. The 8031 does not allow us this luxury, however. While it does have timers (counters, really) that will provide interrupts, the user must reload them between each interrupt to start up the next cycle.

The code developed for this project is heavily commented and is available to download from the Circuit Cellar BBS. I encourage you to download the code and read it over to get a better feel for the way the routines are set up. While using hardware to perform common tasks such as serial communication has the advantage of making the programmer's life easier, sometimes the cost in board real-estate and components can be lowered by the use of cheaper (albeit more complex) software. ■