CIRCUIT CELLAR INK®

THE COMPUTER APPLICATIONS JOURNAL

#59 JUNE 1995

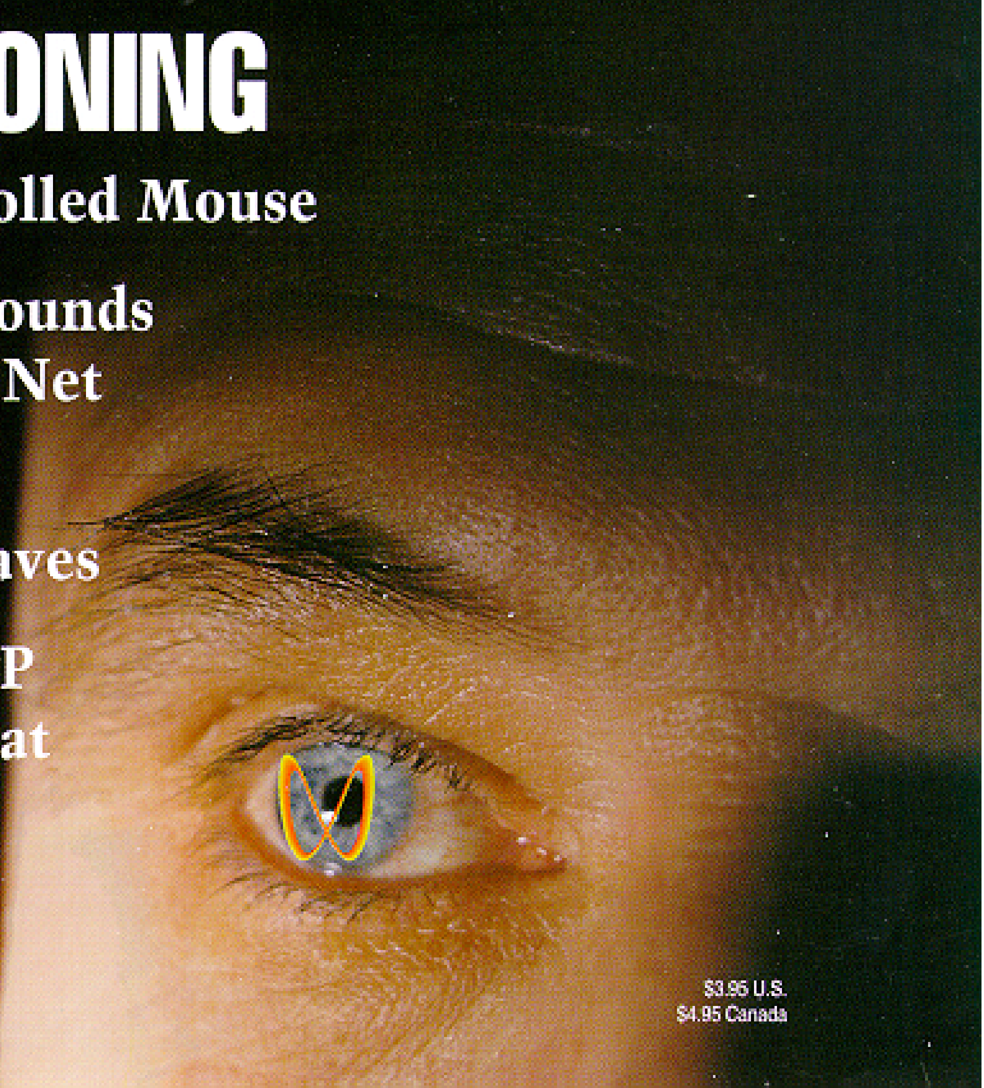
SIGNAL CONDITIONING

An Eye-Controlled Mouse

Recognizing Sounds with a Neural Net

Lancaster's Magic Sine Waves

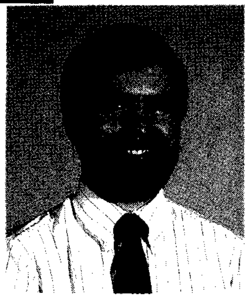
National's COP Still on the Beat



$3.95 U.S.
$4.95 Canada

EDITOR'S INK

Breaking Down the Barriers



I'm convinced that one of the most important tools to come along in the past few years for disabled persons is the computer. No longer must the severely disabled be completely dependent on others. Couple an input device that takes advantage of the mobility they still possess with an on-line connection and their barriers to the outside world are suddenly removed.

I'm sure many of you who frequent on-line services (whether it be a commercial service, BBS, or the Internet) have met a disabled person. You initially assume that they are like the majority of the population: able bodied with full control of their limbs and capable of seeing and hearing the world. Only after you'd developed an appreciation for who they are did you discover (perhaps through something they wrote in a message) that they couldn't see or move their limbs.

Disabled people must overcome obstacles every day of their life. By replacing or supplementing a dysfunctional body part with a computer, they are free to express themselves, less impeded by physical barriers. It may take them longer to write a message, but it doesn't take you longer to read it.

Blind people have speech synthesizers. Deaf people see the screen. Even those who can't fully move their arms frequently have enough mobility to control a joystick. Someone who is paralyzed from the neck down has sip-and-puff devices, however such devices can be somewhat limited. Is there something better?

For these people, we present "The Eye Mouse," the first-place winner of the 1994 Circuit Cellar Design Contest. With this remarkably inexpensive input device, you only need be able to move and blink your eyes to do everything on a computer that can be done with a mouse. Keeping the cost down is an important consideration when designing such devices, and our contest winners did an admirable job.

Along a similar line, another of our feature articles presents a method for recognizing sounds using a neural network. While the implications for the deaf are far less significant than those of the Eye Mouse for a paraplegic, the results are promising.

If you're doing any kind of design work for disabled people, and would like to write it up (or submit it to this year's design contest), drop me a note. I'd love to hear from you.

[illegible]

CIRCUIT CELLAR INK®

THE COMPUTER APPLICATIONS JOURNAL

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

EDITOR-IN-CHIEF
Ken Davidson

TECHNICAL EDITOR
Janice Marinelli

ENGINEERING STAFF
Jeff Bachiochi & Ed Nisley

WEST COAST EDITOR
Tom Cantrell

CONTRIBUTING EDITOR
John Dybowski

NEW PRODUCTS EDITOR
Harv Weiner

ART DIRECTOR
Lisa Ferry

PRODUCTION STAFF
John Gorsky
James Soussounis

CONTRIBUTORS:
Jon Elson
Tim McDonough
Frank Kuechmann
Pellervo Kaskinen

PUBLISHER
Daniel Rodrigues

PUBLISHER'S ASSISTANT
Sue Hodge

CIRCULATION MANAGER
Rose Mansella

CIRCULATION ASSISTANT
Barbara Maleski

CIRCULATION CONSULTANT
Gregory Spitzfaden

BUSINESS MANAGER
Jeannette Walters

ADVERTISING COORDINATOR
Dan Gorsky

CIRCUIT CELLAR INK, THE COMPUTER APPLICATIONS JOURNAL (ISSN 0896-8985) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 875-2751. Second class postage paid at Vernon. CT and additional offices One-year (12 issues) subscription rate U.S.A and possessions $21 95, Canada/Mexico $31 95, all other countries $49.95. All subscription orders payable in US. funds only. via international postal money order or check drawn on U.S. bank. Direct subscription orders and subscription related questions to Circuit Cellar INK Subscriptions, P 0 Box 696, Holmes. PA 19043-9613 or call (600) 269-6301

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept P 0 Box 698, Holmes, PA 19043.9613.

Cover photography bv Barbara Swenson
PRINTED IN THE UNITED STATES

HAJAR ASSOCIATES NATIONAL ADVERTISING REPRESENTATIVES

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

SOUTHEAST
Christa Collins
(305) 966-3939
Fax: (305) 985-8457

MIDWEST
Nanette Traetow
(708) 357-0010
Fax: (708) 357-0452

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

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

NEW PRODUCT NEWS

Edited by Harv Weiner

16-MEGABIT NVSRAM

Benchmarq has announced **bq4017** 16-Mb module, the next-generation of nonvolatile static RAM (NVSRAM). Organized as 2 Mb x 8, the bq40 17 has all the performance characteristics of conventional SRAM with the added benefit of data retention in the absence of power. Because of its performance and density, the module consolidates system memory by replacing several different types of memory in an application.

The bq4017 integrates four 4-Mb static RAMs with two lithium coin cells and a power-control chip to form a nonvolatile reprogrammable memory. The module operates from a single 5-V supply and interfaces like a conventional static RAM with an access time of 70 ns. When the supply voltage drops, the bq4017 automatically write protects the memory and switches power to the internal back-up cells. The long-life lithium cells maintain data in the SRAMs until power becomes valid again. Memory can be retained in the absence of system power for at least 5 years.

Unlike other types of nonvolatile reprogrammable memory, the bq40 17 has no limits on the number of writes that can be performed. The module offers fast read and write cycle times.

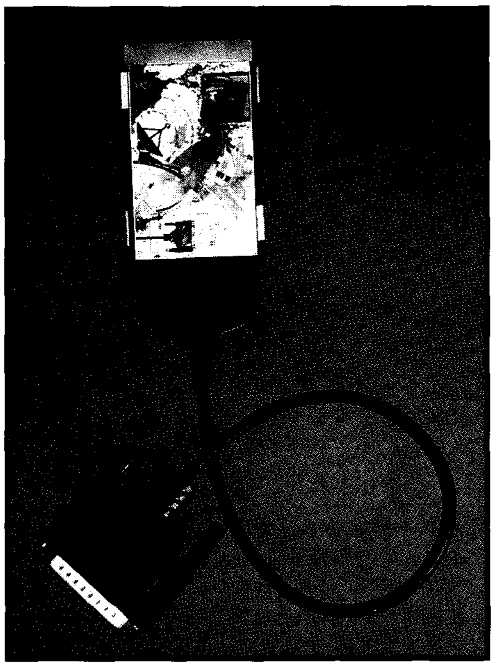
The bq4017 is packaged in a 36-pin, 600-mil DIP that includes the back-up cells. The memory module is completely sealed with a specialized epoxy to protect the cells from heat, solder, moisture, contaminants, and accidental discharge. To ensure the longest possible cell life, the batteries are electrically isolated from the memory until the first application of valid power.

The bq4017 is priced at $900 in quantities of 100.

Benchmarq Microelectronics, Inc.
17919 Waterview Pkwy.
Dallas, TX 75252
(214) 437-9195
Fax: (214) 437-9198

#500

PCMCIA COMMUNICATION CARD



Sealevel Systems has announced a line of PCMCIA serial communication cards. Four PCMCIA Type II cards are available, including **asynchronous RS-232** and RS-4221485, and **synchronous RS-232** and **RS-422/485.** Sealevel is the only manufacturer of the synchronous PCMCIA cards.

The asynchronous cards are based on a 16550 UART, providing compatibility with standard off-the-shelf communication software. The synchronous cards use a single-channel Zilog 85230 Serial Communications Controller suitable for high-speed applications and custom development.

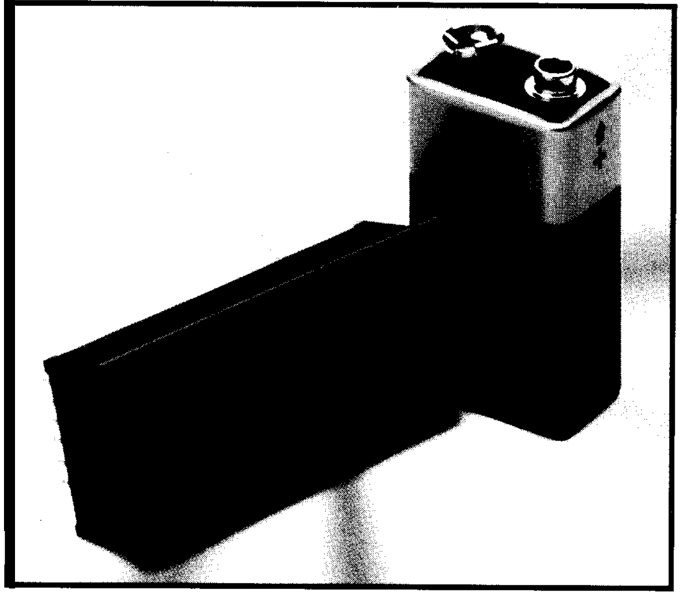
All cards include a high-quality interface cable terminating in a DB-25 connector and all cards conform to the JEIDA 4.1 specification for PC cards.

Pricing for the cards begins at $199.

Sealevel Systems, Inc.
P.O. Box 830 • Liberty, SC 29657
(803) 843-4343 • Fax: (803) **843-3067**

#501

NEW PRODUCT NEWS



LED PANEL METER

Datel has announced a series of 3½-digit, LED-display, digital-panel voltmeters which operate on extremely low power. The **DMS-30PC-RL** series replaces hard-to-read LCD meters or low-power LED meters whose displays are only ¼″ high and whose current drains are in the 40–50-mA range.

The DMS-30PC-RL meters incorporate full-size (0.56" high) low-power LEDs that can easily be read from 20' away. They draw 10 mA from a single +5-V supply and their power consumption is only 50 mW. The meters are housed in rugged, epoxy-encapsulated, component-like 12-pin DIP packages that measure 2.2" x 0.92". The packages, approximately ½″ deep, incorporate a built-in color filter and bezel. The meter's internal A/D converter is a precision autozeroing device that operates from a factory-trimmed reference and guarantees ±1 count accuracy. A Display Test function is standard on each device.

The DMS-30PC-RL series meters are available in three differential-input voltage ranges (±200 mV, ±2 V, and ±20 V). The high-impedance (up to 1000 MΩ) inputs are overvoltage protected to ±250 V and boast a common-mode rejection of 86 dB. All models feature autopolarity changeover and overrange indication.

For popular applications (4-20 mA, RMS-to-DC conversion, AC-line power, J and K thermocouples, etc.], the DMS-30PC-RL series includes a complete line of plug-on applications that conveniently convert the meter into an application-specific instrument.

Prices for the DMS-30PC-RL Series meters start at $49.

Datel, Inc.
11 Cabot Blvd.
Mansfield, MA 02048
(508) 339-3000
Fax: (508) 339-6356

#502

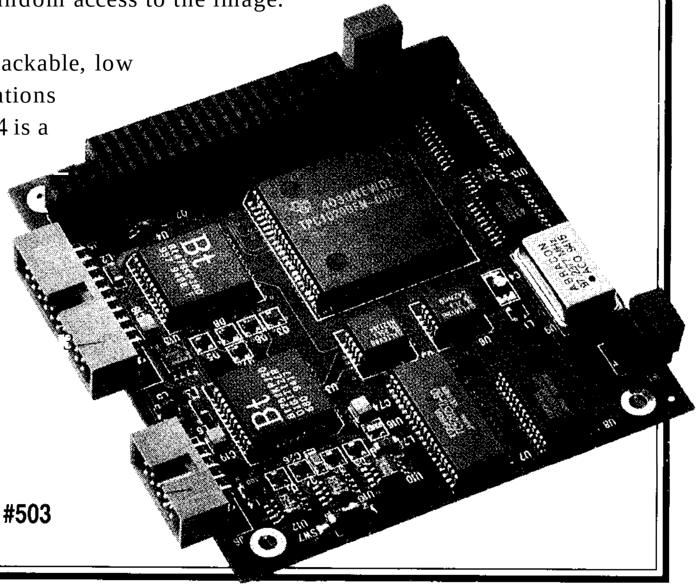
FRAME GRABBER

The **CX104** Frame Grabber from ImageNation offers outstanding image quality with a sampling jitter of ±3 ns and video noise less than one LSB. The image is captured to memory-mapped, dual-ported video RAM to provide fast random access to the image. The image transfer rate is 1 MBps.

The PC/104 bus is the familiar ISA bus in a compact, stackable, low power, 3.6" x 3.8" format. This is ideal for embedded applications that are software compatible with standard PCs. The CX104 is a +5-VDC-only design that draws as little as 5 mA.

A software interface, C library with source code, and Windows DLLs for both C and Visual Basic developers are provided to speed application development. Documentation includes source code and examples for many common image-acquisition needs,

The CX104 family of precision frame grabbers start at **$350** in quantity.

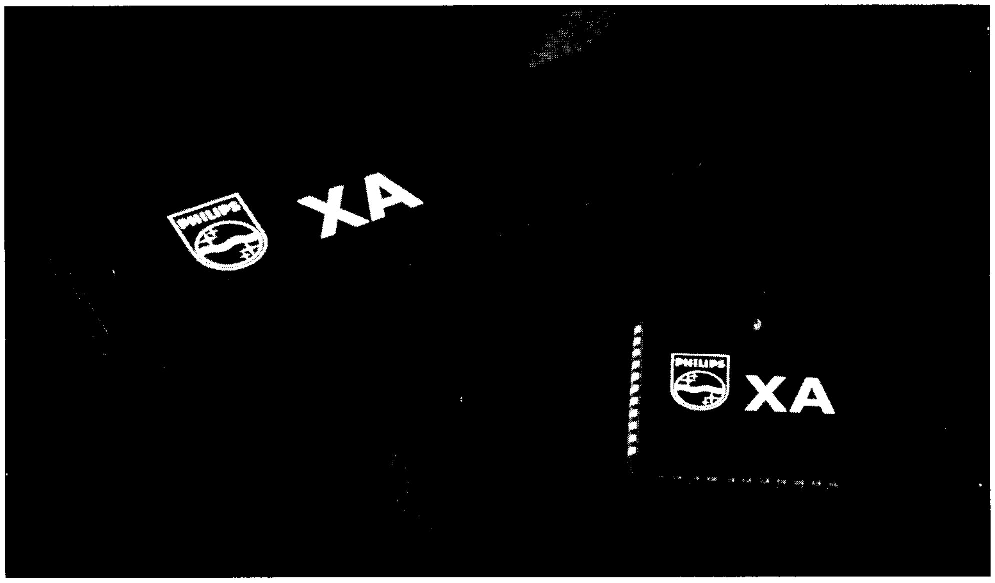


ImageNation Corp.
P. 0. Box 276 • Beaverton, OR **97075-0276**
(503) 641-7408 • Fax: (503) 643-2458

#503

NEW PRODUCT NEWS

16-BIT MICROCONTROLLER



Philips Semiconductors announces the **XA-G3**, its first XA (extended Architecture) derivative CMOS 16-bit microcontroller. The XA-G3 provides upward compatibility for S-bit 80C51 users who need higher performance. The XA-G3 is source-code compatible with the 80C5 1 enabling products to migrate up to 16-bit power without large software and personnel investments.

Any 80C5 1 instruction can be translated directly into one XA instruction. The XA memory map is a superset of the 80C5 1 memory map and all of the 80C5 1 memory addressing modes are supported. Translation programs can use model files to resolve references to special function registers which may be located in different places on different 80C5 1 family and XA derivatives.

Special features of the XA-G3 include 20-bit address range with 1 MB for each program and data space, 32 KB of on-chip EPROM/ROM program memory, 512 bytes of on-chip data RAM, three standard 16-bit counter/timers with enhanced features, watchdog timer, two enhanced UARTs, four S-bit I/O ports with four programmable output configurations, and 2.7–5.5-V operation. By reengineering the design of the 8-bit microcontroller to **16** bits, the XA-G3 is 10–100 times faster than the fastest 8-bit 80C51.

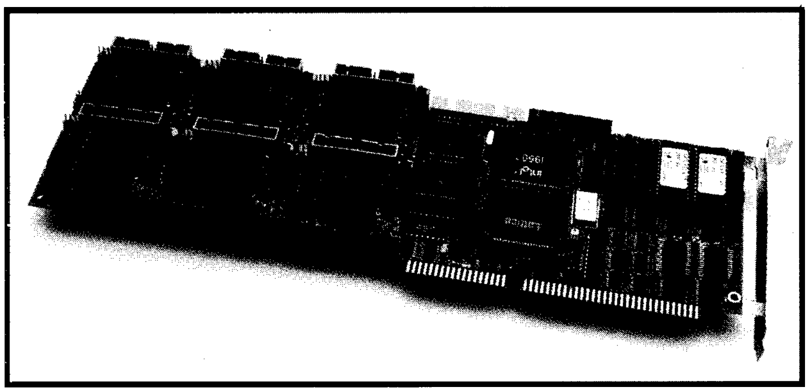
Philips Semiconductors
811 E. Arques Ave.
Sunnyvale, CA **94088-3409**
(408) 991-5192
Fax: (408) 991-3773

#504

MOTION CONTROLLER

Precision MicroControl has introduced the DCX-AT, a state-of-the-art digital motion and I/O controller for installation in an AT/ISA bus or for stand-alone operation. Onboard capabilities include multitasking, complex contouring, S-curve velocity profile, and continuous path motion with cubic-spline interpolation.

The DCX-AT features a modular architecture and can be configured to provide l-6 axes of servo and/or stepper motion control, with 26 dedicated I/O lines for each axis of control, and 16-96 undedicated digital I/O lines. Multiple DCX-AT controllers can be joined to provide up to 96 axes of control and 1536 undedicated I/O lines.

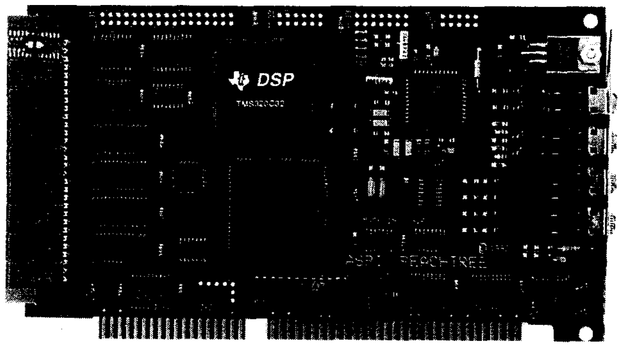


The DCX-AT is supplied with clear, concise manuals for installation, setup, programming, and operating. Software for creating user interfaces and writing and executing motion programs is included.

Precision **MicroControl** Corp.
2075-N **Corte** del **Nogal**
Carlsbad, CA 92009
(619) 930-0101 • Fax: (619) 930-0222

#505

NEW PRODUCT NEWS



LOW-COST DSP PLATFORM

The **Peachtree** DSP **Platform** from ASP1 is a low-cost OEM board featuring the Texas Instruments TMS320C32 Digital Signal Processor running at 40 MFLOPS. The TMS320C32 offers the ease of use and performance of 32-bit floating-point DSPs as well as the cost advantage of 16-bit fixed-point DSPs. It is source- and object-code compatible with the TMS320C3*x* family and source-code compatible with the TMS320C4*x* family, providing a lower cost floating-point DSP alternative.

The Peachtree DSP Platform also includes a stereo 16-bit A/D and D/A converter and 32 kilowords (128 KB) of zero wait-state static RAM. A byte-wide DRAM interface is provided for low-cost data memory expansion up to 16 MB. The TMS320C32's external memory interface can automatically transfer 8-, 16-, or 32-bit quantities into and out of memory. The values are converted into an internally equivalent 32-bit representation.

The Peachtree DSP Platform is available in a Software Developer's Kit which contains the TI TMS320 floating-point DSP optimizing C compiler, TMS320 floating-point DSP assembly language tools, and a C source debugger. Also included is ASPI's real-time I/O system, which provides memory management, real-time device-independent drivers, and a host-to-DSP communications system.

Applications that could run on the platform include speech compression and analysis, audio data acquisition and processing, vibration and noise analysis, acoustics research, music synthesis, speech recognition, text-to-speech, and other audio applications.

The Peachtree DSP Platform sells for $595.

Atlanta Signal Processors, Inc.
1375 Peachtree St. NE,
Ste. 690
Atlanta, GA 30309-3115
(404) 892-7265
Fax: (404) 892-2512

#506

RS-232 TRANSCEIVER

Maxim has introduced two RS-232 transceivers that achieve 1-µA supply current with an AutoShutdown feature. When the MAX3223 and MAX3243 ICs do not sense a valid signal level on their receiver inputs, the onboard power supply and driver shut down, reducing supply current to 1 µA. The system turns on again when a valid level is applied to any RS-232 input. As a result, the system saves power without changes to the existing BIOS or operating system.

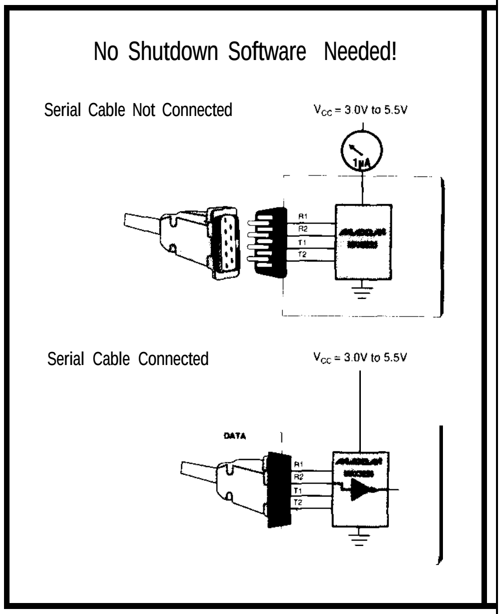
A proprietary, high efficiency, dual-charge-pump power supply and a patented low-dropout transmitter combine to deliver true RS-232 performance using 3.0–5.5-V supplies. A guaranteed data rate of 120 kbps provides compatibility with popular software for communicating with personal computers.

The MAX3243 3-driver and 5-receiver complete serial port is ideal for notebook computers (or equivalent). Besides having all receivers active in shutdown, one receiver has a second, complementary output that is always active. This receiver monitors an external device (such as a modem) in shutdown, without forward biasing the protection diodes in a UART that may have Vcc completely removed.

The MAX3223 is available in a 20-pin DIP and SSOP packages while the MAX3243 comes in 28-pin wide SO and SSOP packages. Prices for the MAX3223 start at $1.85 per 1000.

Maxim Integrated Products
120 San Gabriel Dr. • Sunnyvale, CA 94086
(408) **737-7600** • Fax: (408) 737-7194

#507



FEATURE ARTICLE

Seibert L. Murphy &
Samir I. Sayegh

Artificial Neural Network Recognizes Sounds

It's one thing to hear sound. It's quite another to intelligently identify the sound. Here, neural nets do just that—they recognize a tone whether it's generated by a signal generator or a loudspeaker.

Artificial neural networks, neural networks, or Nets (NNs) are '90s buzz words. With fast, inexpensive computers, almost anyone can experiment with their own version of NNs.

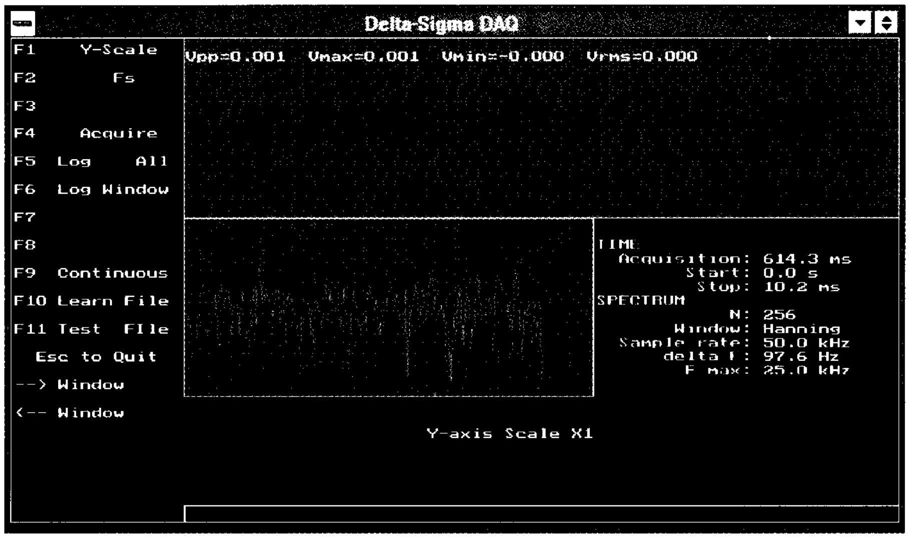
We decided to construct an "artificial ear" capable of distinguishing different sounds. We wanted to train a NN to recognize three types of tones directly from a signal generator and then indirectly from a loudspeaker. This investigation involved integrating components and software as well as formulating training and tests of the artificial intelligence.

THE HUMAN ARCHETYPE

As humans, we can recognize different kinds of sounds with our ears and the auditory processing centers in our brain. We process time and amplitude fluctuations to extract pitch and timbre and make quality judgments about what we hear.

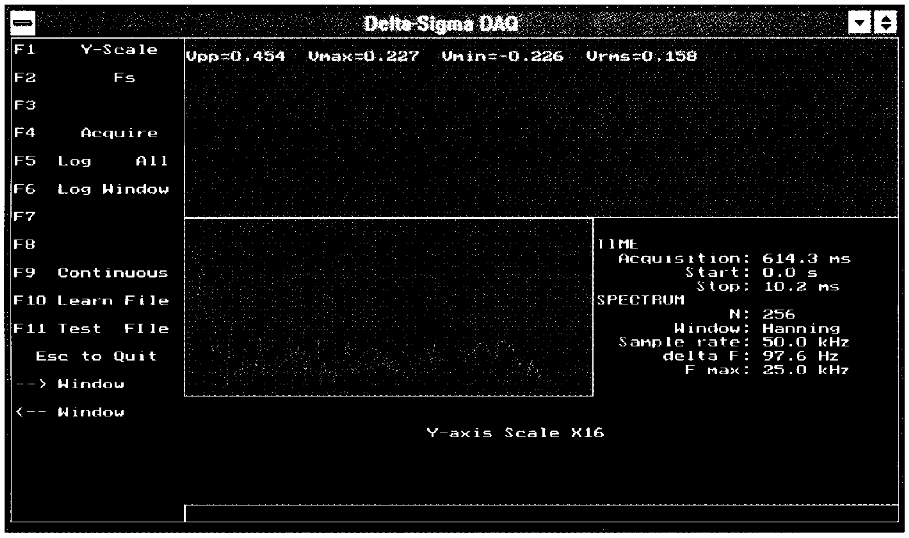
In this experiment, the computer needs to recognize different sound signals-both discrete and combined. We programmed a PC to turn on an output to any of three channels, depending on how simple or complex the sound was.

Having a computer recognize sounds is not new, but doing it with



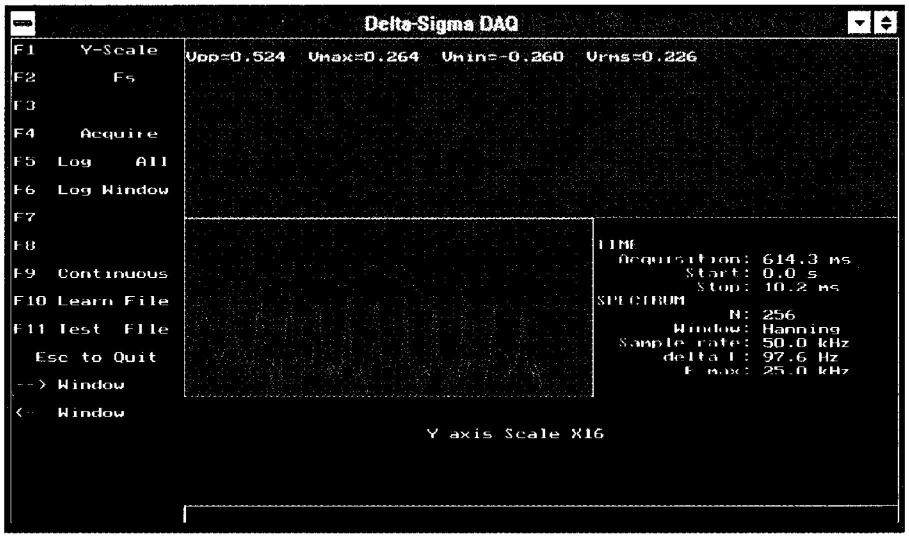
```
Vpp=0.001  Vmax=0.001  Vmin=-0.000  Vrms=0.000

TIME
  Acquisition: 614.3 ms
        Start: 0.0 s
         Stop: 10.2 ms
SPECTRUM
            N: 256
       Window: Hanning
  Sample rate: 50.0 kHz
      delta F: 97.6 Hz
        F max: 25.0 kHz

Y-axis Scale X1
```

Figure 1—*NDAQ allows* you to simultaneously view time *domain (top) and frequency* domain *(bottom)* traces. The *function key menu provides easy control over data acquisition modes and storage options.*



```
Vpp=0.454  Vmax=0.227  Vmin=-0.226  Vrms=0.158

TIME
  Acquisition: 614.3 ms
        Start: 0.0 s
         Stop: 10.2 ms
SPECTRUM
            N: 256
       Window: Hanning
  Sample rate: 50.0 kHz
      delta F: 97.6 Hz
        F max: 25.0 kHz

Y-axis Scale X16
```

Figure 2-A *sinusoidal* time domain *trace indicates that this is a sine wave. The spectrum of a sine wave is characterized by a sing/e predominant peak* in *the frequency domain, in this case 1,000 Hz.*



```
Vpp=0.524  Vmax=0.264  Vmin=-0.260  Vrms=0.226

TIME
  Acquisition: 614.3 ms
        Start: 0.0 s
         Stop: 10.2 ms
SPECTRUM
            N: 256
       Window: Hanning
  Sample rate: 50.0 kHz
      delta F: 97.6 Hz
        F max: 25.0 kHz

Y axis Scale X16
```

Figure 3-A *square wave is characterized by ifs square leading and trailing edges. Unlike the sine wave, the spectrum is full of odd-order harmonics. Perceptually, the presence of odd-order harmonics makes the sound seem harsh,* like a foghorn.

NNs is. By using NNs, however, the decision is continuous (i.e., its output closely resembles the analog judgment of an human observer).

The ability of the NN to make fuzzy judgments gives us more intuitive answers. The network can determine, for instance, that a particular signal very strongly fits category 1 and to some degree fits category 2.

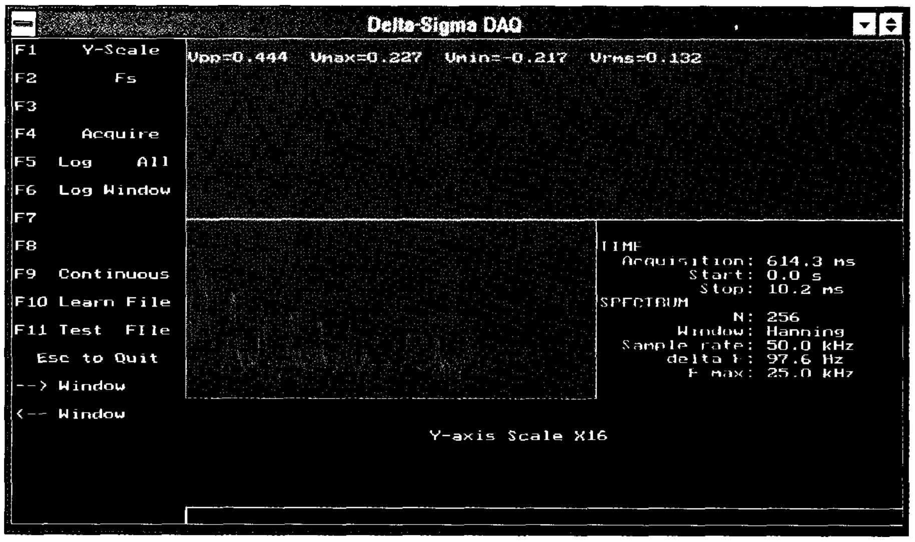
In our experiment, we asked the NN to characterize sine, square, and triangular waves by applying an analog level to channel 1, 2, or 3, respectively. In the case of a sine wave, the NN was trained to apply a 1.0 to channel 1 and -1.0 to channels 2 and 3.

In a real situation, the NN could then tell us whether the input signal was a specific type or combination of wave patterns. This feature enables a NN to decide whether an acceptable level of distortion is present in a signal or to detect specific features.

NN'S APPLICATION

NNs provide a theoretically sound approach to solving a variety of engineering and scientific problems considered traditionally difficult. While an exact definition remains elusive (practitioners tend to emphasize one or another NN characteristic), it is possible to list the most common and fundamental features of NN solutions.

- Adaptive-Adaptive solutions are more stable and manage huge data sets with minimal memory requirements since the patterns are presented one at a time. This advantage enables real-time, on-line solutions, where the totality of the data set is not available at the outset.
- Parallel—NN solutions are inherently parallel. A large number of nodes share computation, often operating independent of other nodes. This method speeds computation and enables implementation on some highly efficient parallel hardware.
- Neurally inspired—There are some similarities between current artificial neural algorithms and biological systems capable of intelligence. The fact that such biological systems



```
Delta-Sigma DAQ
F1    Y-Scale     Vpp=0.444  Vmax=0.227  Vmin=-0.217  Vrms=0.132
F2        Fs
F3
F4    Acquire
F5    Log    All
F6    Log Window
F7
F8                                  TIME
                                       Acquisition: 614.3 ms
                                             Start: 0.0 s
                                              Stop: 10.2 ms
F9    Continuous                    SPECTRUM
F10 Learn File                                   N: 256
F11 Test  File                              Window: Hanning
  Esc to Quit                          Sample rate: 50.0 kHz
--> Window                                delta f: 97.6 Hz
<-- Window                                  F max: 25.0 kHz
                    Y-axis Scale X16
```

Figure 4-*The regular triangular pattern of the triangle wave is seen in the fop trace. The frequency domain pattern is c/ear/y different from either the* **square** *or sine wave. The presence of harmonics* in *the lower ha/f of the spectrum* gives *a rich tonal* qualify.

still display pattern-recognition capabilities far beyond those of our algorithms is a continuing incentive to maintain and further explore the neurobiological connection.

- Ability to handle nonlinear problems transparently-This ability is fundamental to modern science and engineering. In a number of fields, nonlinear approaches are developed on a case-by-case basis and with little connection to better established linear techniques.

  However, by formulating a NN and endowing it with increasingly complex processing capabilities, we can define a unified spectrum from linear networks (e.g., a one-weight-layer ADALINE) to highly nonlinear ones with powerful processing capabilities (e.g., a multilayer back-propagation network).

These properties, coupled with a nearly universal model of NNs and the availability of software and hardware tools, make NNs one of the most attractive instruments of signal processing and pattern recognition available today.

A NN MODEL

Selecting the best NN topology was the most time-consuming part of the project. We experimented with several topologies and settled on the following:

Number of layers: 3 (1 input, 2 hidden, and 1 output layer]
Inputs: 256
Hidden Layer 1: 3 1 nodes
Hidden Layer 2: 4 nodes
Output Layer: 3 nodes

Training was accomplished by back-propagation, a systematic, mathematically sound, robust method of training a net. You train a net to generate a set of connection weights, which then generates a desired output pattern when it is applied to the input. Backpropagation calculates weights by feeding errors backwards though the net and adjusting the weights to minimize errors.

HARDWARE AND SOFTWARE PARAMETERS

Deux Ex Machina's (DEM) précis board ***(INK 49)*** was chosen as our A/D converter. At less than $400, precis is reasonably priced and provides a sigma-delta converter, 16-bit resolution, and 100-kHz sampling rate. The data acquisition program (DAQ) is menu driven, enabling you to control the system using function keys.

We were able to digitize 30,720 points during a single acquisition. Our program, shown in Figure 1, includes a menu bar that allows selection of sample rate, vertical scale sensitivity, and logging mode.

InfoTech's DynaMind V3.0 NN development software was also used. This midpriced NN builder includes the basics needed to fully evaluate an application. Its GUI sells for $195 and has excellent documentation as well as tutorial and example programs.

(The full-blown developer kit sells for $1795 and includes a GUI to build, train, and test NN models and a C code generator to embed the final network in your own application!)

The project ran on a 486/33 computer configured with 16 MB of RAM. Code supplied by DEM was ported to our software package. Although the programs can run in

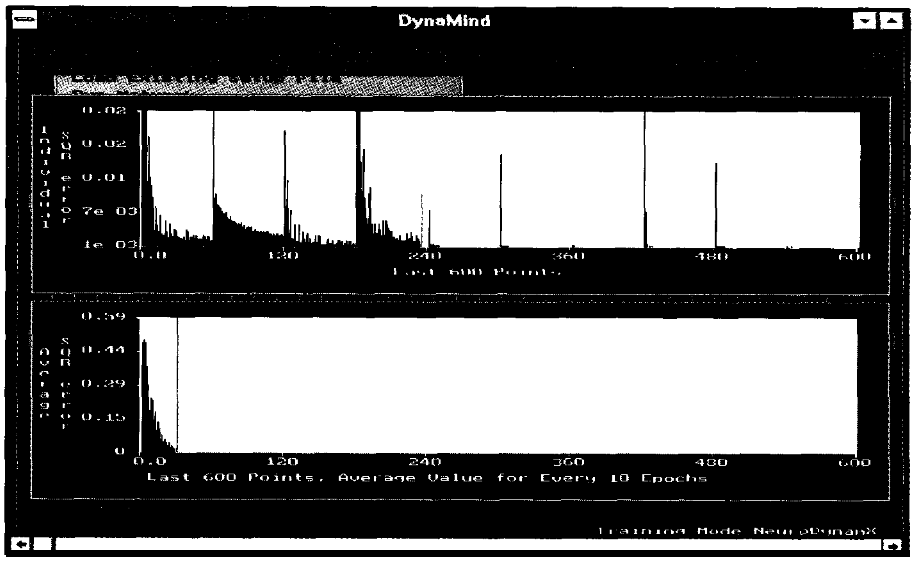


Figure **5**—*Real-time* progress updates of the *NN's* performance are presented in *two* formats. The *fop* trace shows *the current "epoch" performance. The average square error of the back-propagation process is displayed on the bottom trace and is updated every 10 epochs.*

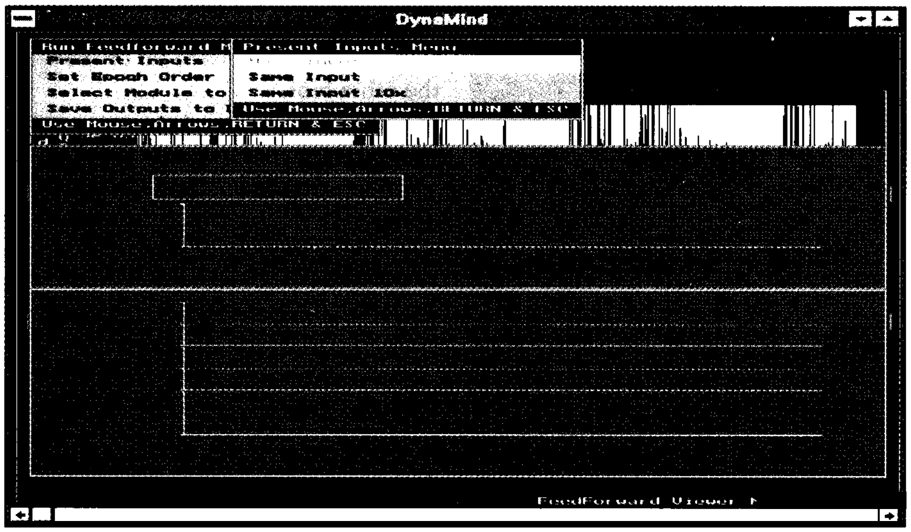


Figure 6—*New* input patterns can *be compared to the computer "learned" results in the feedforwad display.*

conventional memory, a minimal configuration of a 386/33 with 8 MB RAM is best. The NN uses the additional memory to store network and data records.

SIGNALS

The program we developed is configured so users can select one of three output channels, labeled Type 1–3. The user arbitrarily assigns the signal type to any of the three channels. Our output channel assignment is mapped to the type of stimulus signal. For this experiment, we assigned the following channels as:

- Type 1: Sine
- Type 2: Triangular
- Type 3: Square

The précis board's default input range is ±1 V p-p. We chose a signal level of 100 mV RMS (0.1414 V p-p) to give plenty of head room and a stimulus frequency of 1000 Hz.

The choice of the stimulus signals was based on the way each one sounds.

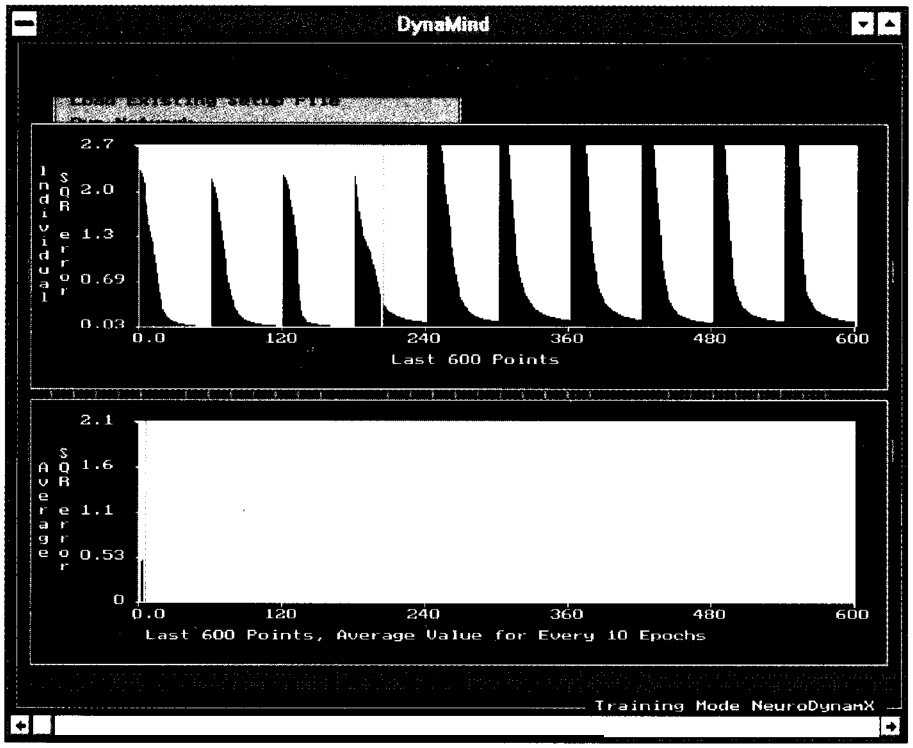


Figure 7a—*At* the *beginning of the training session (52s), the individual epoch square errors are in excess of 2.7, indicated by the red.*

FREE

Data Acquisition Catalog

1995

PC and VME data acquisition catalog from the inventors of plug-in data acquisition. Featuring new low-cost A/D boards optimized for Windows, DSP Data Acquisition, and the latest Windows software. Plus, informative technical tips and application notes. Call for your free copy

1-800-648-6589

ADAC

American Data Acquisition Corporation
70 Tower Office Park, Woburn, MA 01801
phone 617-935-3200 fax 617-938-6553

#105

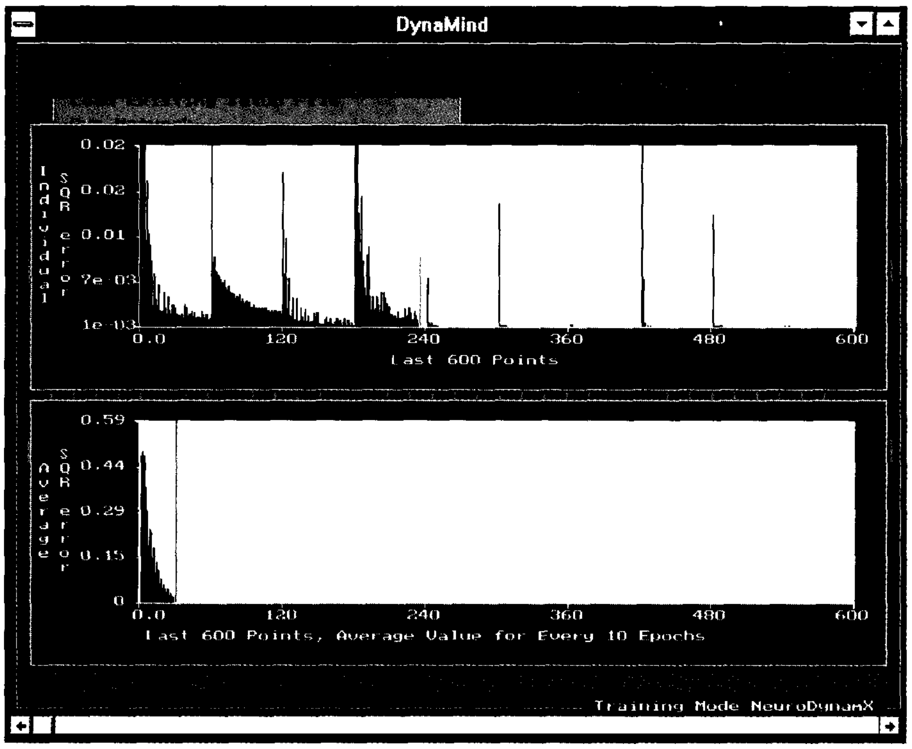


Figure 7b—*After* 1.5 h, the *individual square* error is typical*ly under 0.02. The* occasional *red* bands indicate *short periods of errors in judgement,* but the network quickly adjusts itself.

To the ear, each wave shape has a distinct tonality. A pure sine signal has no harmonics other than the fundamental tone. As you can see in Figure 2, it sounds discrete, like a note played on a flute.

A square wave, on the other hand, exhibits strong harmonics, specifically the fundamental and odd-order harmonics. Figure 3 indicates that the sound is harsh like a foghorn.

The triangular wave contains odd harmonics that decay quickly with increasing frequency. Aurally, the sound has a rich quality (see Figure 4).

OUTPUT FROM DAQ AND NN

Raw data sets are time-domain samples collected during the acquisition interval. To make the signal classification, the NN uses frequency-domain information. We preprocessed each of the time samples with a DC filter and a Hanning window function before performing the FFT and power-spectrum calculations.

The DC filter removes any DC component that may be present. During training, we found that DC levels tend to throw off the network, which is probably due to the NN's sensitivity to absolute levels.

Large, variable DC levels expose any anomalies in the data. In some cases, the contribution of this first bin data (f0) to the NN weights saturates the network, forcing output levels to ± 1. As a result, the network takes a long time (or may not be able) to organize correctly based on the data in the other bins.

The Hanning window minimizes adjacent bin crosstalk. The shape of the Hanning window provides a good balance between level and frequency resolution. Our hunch that relative levels are slightly more important than precise frequency was confirmed when the same network successfully recognized signals whose frequencies were ±10% of the training frequency.

PROCESSING

Using Nyquist's theorem, we determined that we could only obtain an FFT spectrum of 25 kHz since précis was set up to sample at 50 kHz.

Our time record length is based on the number of independent samples and the size of the sample record. In a sample size of 512 points, we collect 60 records per acquisition. This offers a total record length of:

$$T = 60 \times \frac{512}{50}\ \text{kHz} = 614\ \text{ms}$$

and an FFT bin width of:

$$df = \frac{25}{256}\ \text{kHz} = 97.7\ \text{Hz}$$

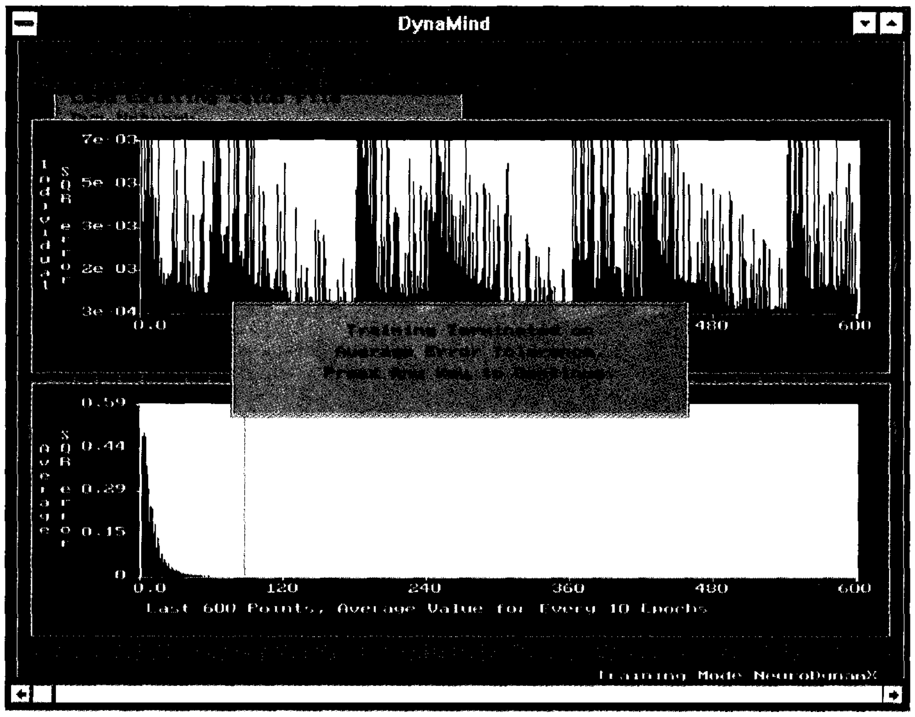

DynaMind
Individual SQR error
7e-03
5e-03
3e-03
2e-03
3e-04
0.0
480
600
Training Terminated on Average Error Tolerance. Press Any Key to Continue.
Average SQR error
0.59
0.44
0.29
0.15
0
0.0
120
240
360
480
600
Last 600 Points, Average Value for Every 10 Epochs
Training Mode NeuroDynamX

Figure 8—*Once the goal of an average square error of less than 0.005 is attained, the software automatically displays fhe termination screen. This allows you to go off and do other tasks without having to babysit the NN's learning process.*

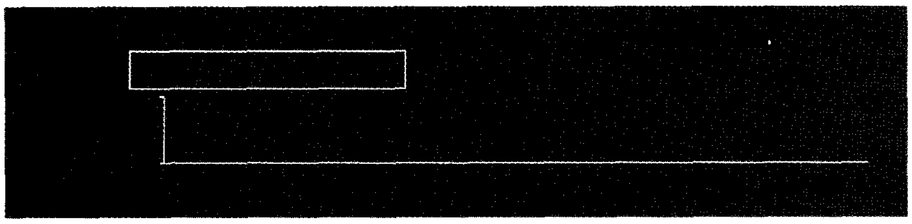


Figure 9-*The input pattern in fhis case is a sine wave. Although the data was scaled, a single peak in the frequency domain is clearly seen.*

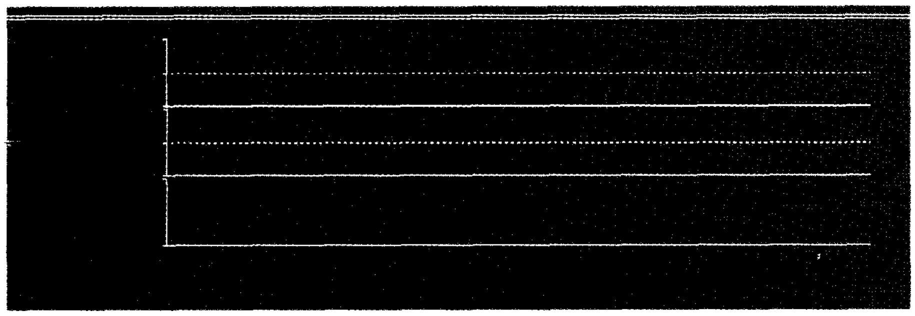
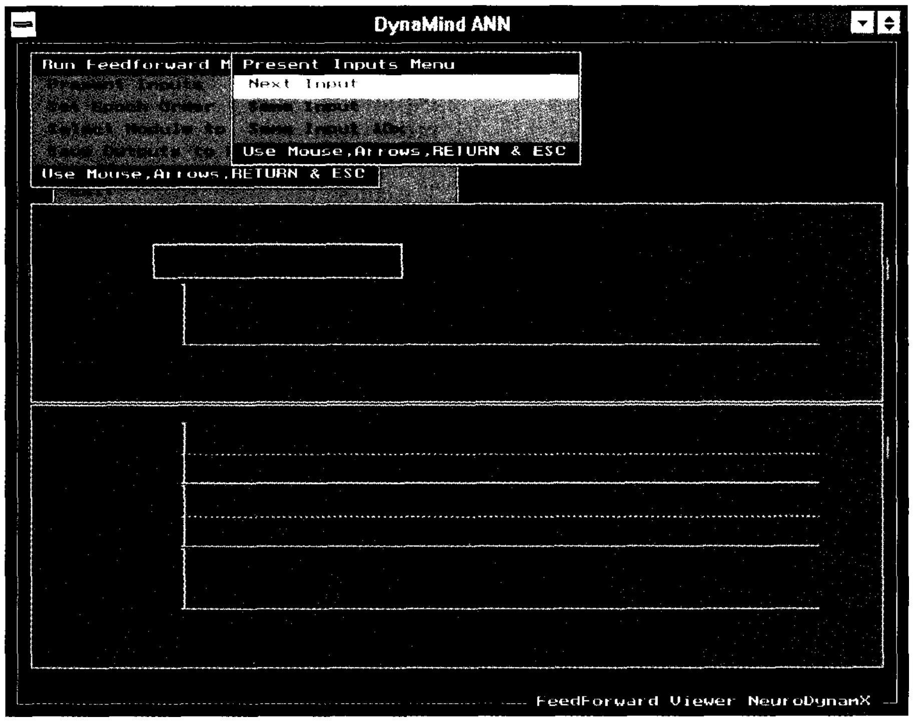


Figure 10—*The ability of the NN to very accurately identify the input pattern is graphically displayed. The classification pattern made by the Net is displayed as the "Output." While the classification made by fhe "human" is displayed as the "Target. " A graphical difference display "Error" shows that we are in agreement!*

During data acquisition and training, there are two major things to look for: consistency and as wide a representative group of samples as possible. Although this sounds contradictory, let me explain.

Consider the untrained NN a jumbled puzzle. The training session mathematically adjusts connections between the various layers so the computations performed on the input pattern give the desired output pattern (i.e., it provides supervised learning).

The final weight values make the connection map that shows how the puzzle fits together. Like successive approximation, the network adjusts weights after each update using back-propagation until errors are eliminated or reduced to an acceptable level.

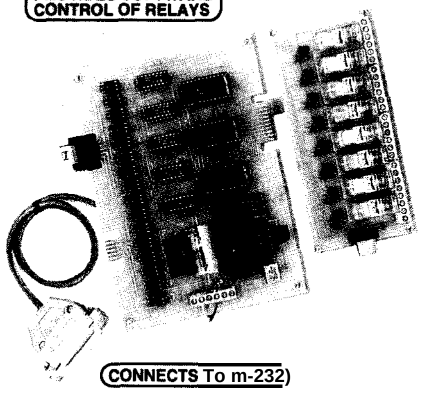


DynaMind ANN
Run Feedforward M
Present Inputs Menu
Next Input
Use Mouse,Arrows,RETURN & ESC
Use Mouse,Arrows,RETURN & ESC
FeedForward Viewer NeuroDynamX

Figure 1 l-/n *fhis feed-forward output screen, we see that a square wave is being tested. Note the steady decay of harmonics.*

RELAY INTERFACE

PROVIDES SOFTWARE CONTROL OF RELAYS

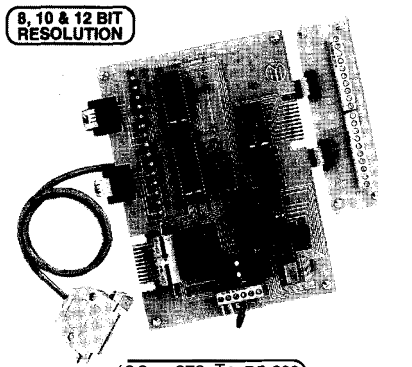


(CONNECTS To m-232)

AR-16 RELAY **INTERFACE** (16 channel)............ $69.95
Two 6 channel (TTL level) outputs are provided for **connection to relay cards or other devices (expandable to 128 relays using EX-16 expansion cards). A variety of relays cards and relays are stocked. Call for more info.**
AR-2 RELAY INTERFACE (2 relays, 10 **amp)....$** 44.95
RP6 REED RELAY CARD (6 **relays,** 10 VA)...... $49.95
RH-8 RELAY CARD **(10 amp SPDT, 277 VAC)...$** 69.95

ANALOG **TO DIGITAL**

8, 10 & 12 BIT RESOLUTION



(CONNECTS To **RS-232**)

ADG16 **A/D** CONVERTER* (16 **channel/8 bit)..$** 99.95
ADC-8G A/D CONVERTER* **(8** channel/l0 **bit).$124.90**
Input voltage, amperage, pressure, energy usage, joysticks and a wide variety of other types of analog signals. RS-422/RS-485 available (lengths to 4,000'). Call for info on other A/D configurations and 12 bit converters (terminal block and cable sold separately).
ADC-BE TEMPERATURE **INTERFACE*** (6 **ch)..$** 139.95
Includes term. block & 6 temp. sensors (-40' to 146' F).
STA-6 DIGITAL INTERFACE' (6 **channel).........$** 99.95
Input on/off status of relays, switches, HVAC equipment. security devices, smoke detectors, and other devices.
STA-8D TOUCH TONE INTERFACE'............... **$** 134.96
Allows callers to select control functions from any phone.
PS-4 PORT SELECTOR (4 channels **RS-422)....$ 79.95**
Converts an RS-232 port into 4 selectable RS-422 ports.
CO-465 (RS-232 to **RS-422/RS-485 converter)......$ 44.95**

***EXPANDABLE...**expand your interface to control and monitor up to 512 relays, up to 576 digital Inputs, up to 126 analog inputs or up to 128 temperature inputs using the PS-4. EX-16, ST-32 & AD-16 expansion cards.

FULL TECHNICAL **SUPPORT...**provided over the telephone by our staff Technical reference &disk including test software & programming examples in Basic, C and assembly are provided with each order.

HIGH **RELIABILITY...**engineered for continuous 24 hour industrial applications with 10 years of proven performance in the energy management field.

CONNECTS TO RS-232, **RS-422** or **RS-485...**use with IBM and compatibles, Mac and most computers. All standard baud rates and protocols (50 to 19,200 baud)

Use our 800 number **to** order FREE INFORMATION PACKET. Technical information (614) 464-4470.

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

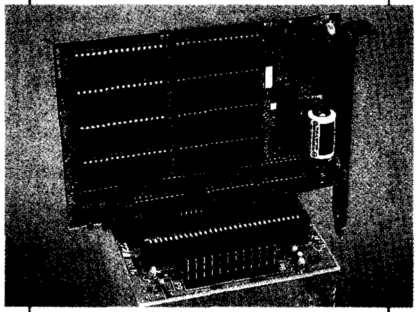
International & Domestic FAX (614) 464-9656
Use for information, technical support & orders.

ELECTRONIC ENERGY CONTROL, INC.
360 South Fifth Street, Suite 604
Columbus. Ohio 43215-5438

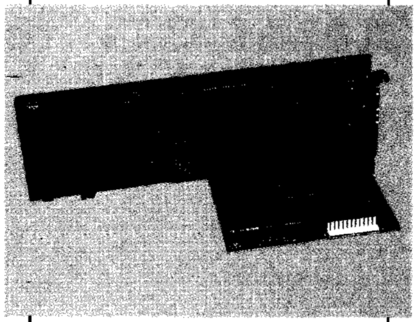
#106

VMAX®

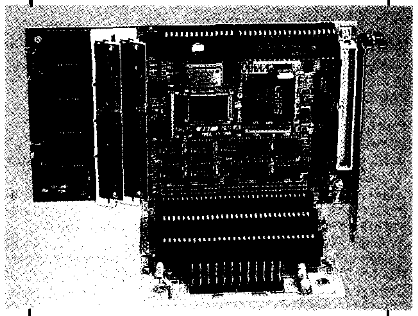
QUALITY PRODUCTS
RESPONSIVE SERVICE
RELIABLE DELIVERY



SOLID STATE DISK —$135*
½ Card 2 Disk Emulator
EPROM, FLASH and/or SRAM
Program/Erase FLASH On-Board
1M Total, Either Drive Bootable



25MHZ 386DX CPU —$695*
Compact AT/Bus or Stand Alone
On-Board SVGA, IDE, FDC, 2 Ser/Bi-Par
FLASH&RAM Drives to 2.5M
Cache to 128K, DRAM to 48M



TURBO XT
w/FLASH DISK —$266*
To 2 FLASH Drives, 1M Total
DRAM to 2M
Pgm/Erase FLASH On-Board
CMOS Surface Mount, 4.2" x 6.7"
2 Ser/1 Par, Watchdog Timer

All Tempustech VMAX products are
PC Bus Compatible. Made in the
U.S.A., 30 Day Money Back Guarantee
*QTY 1, Qty breaks start at 5 pieces.

TEMPUSTECH, INC.
TEL: (800) 634-0701
FAX: (813) 643-4981

Fax for fast response!	295 Airport Road Naples, FL 33942

#107

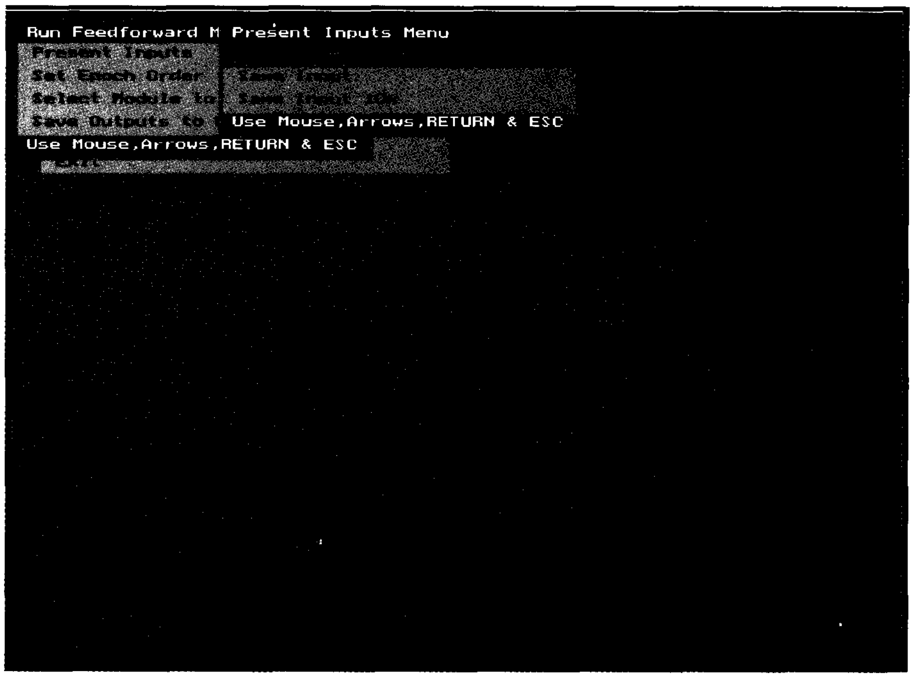

Run Feedforward M Present Inputs Menu
Present Inputs
Set Epoch Order
Select Module to
Save Outputs to
Use Mouse,Arrows,RETURN & ESC
Use Mouse,Arrows,RETURN & ESC

Figure 12—*The* **net** *is processing a triangular* wave reproduced by a loudspeaker. Compare *this to the sine wave in Figure* 6.

We are then required to show the network data that is accurately identified so that previous "correct" weights are reinforced. As well, we need to show as many different patterns as possible so the weights are evenly distributed in the connection space. A network can make gross generalizations on small data sets and can also not converge on data sets that are too large or inaccurate.

DynaMind's GUI enabled us to visualize all aspects of the NN's performance, including weight distribution. The following two modes are most useful:

- Train Network Mode-controls learning parameters and views the training results in real time. In Figure 5, the error is a function of epoch, where an epoch is one complete training cycle of the NN.
- Run Feedforward Mode-tests what the NN "learns." In Figure 6, the predicted output is compared with the NN results. Note theerror reading in the bottom trace of the graph marks a very close correlation.

Other modes enable you to view the weights by layer, choose different activation functions, or change the training paradigm altogether.

We trained the network on data directly from a signal generator so we could feed it precisely controlled time records. In addition, the signal generator's controls enabled us to adjust amplitude and symmetry so we could introduce a wide range of signal structures.

TRAIN AND TEST

The training of a NN takes place in two stages: acquiring data and performing training. Training follows a predictable cycle:

1. Acquire raw learn data
2. Preprocess raw data
3. Set NN parameters
4. Set termination conditions
5. Perform back-propagation learning and update connection weights
6. Exit condition (at this point, you can completely exit the program or repeat step 5)

NDAQ.EXE performs the first two tasks while our program handles digitization and preprocessing. (Preprocessed spectrums are automatically stored as LEARN.10 or TEST.10 when you complete data collection.)

NN topology parameters are initialized for the software by selecting TRAIN.SET from the Load Setup menu. We configured this file with the network parameters described earlier.

Although the back-propagation training method is not the fastest, it is easily represented graphically. You must remember that initial weights must be randomized (something far too easy to forget!) If you train "new" patterns using "old" weights, the NN may never converge on a satisfactory answer. Figures 7a and b depict square-error versus epoch at two times during the training session. Note the automatic y-axis scaling.

Training may take an hour and a half or more, depending on your hardware and graphics mode, so relax. With graphics mode set on, training takes over 3 hours on a 486DX/33.

At the end of the training session, the program displays the termination condition (see Figure 8). Termination is determined by evaluating the epoch's mean-square error. After the minimum error is reached, observe the performance of the network by running in feed-forward mode.

Most graphical NN packages let you set termination conditions based on maximum number of epochs or a minimum average square error. Here, we chose 10,000 epochs or an average square error of less than 0.005.

At the end of this session, the epoch error is at 0.0019. As you can see in Figure 8's bottom graph, NNs exhibit the classic pattern of exponential decay in the error curve.

At this point, you should look at the network performance in detail. We were able to view each spectral pattern presented for training.

Indexing through all of the input patterns lets you observe the error value (see Figure 9) and the output, target, and error graphs (see Figure 10). The output graph shows the NN computed output. The target graph illustrates the desired decision you specified during training. The NN performance is displayed in the error graph at the top of the display and has dimensions of error squared.

USING A TRAINED NN

Testing and using the trained NN also requires the NDAQ software and the following steps:

1. Acquire raw new data
2. Perform feed-forward test
3. Evaluate performance using the feed-forward screen

For testing, select TEST. IO (the same TRAIN. SET setup is used).

Remember, the goal of your effort is to get the NN to make decisions similar to your own. Perform the test cycle a number of times to ensure the network is performing satisfactorily. The output of the feed-forward screen is shown in Figures 9-11.

A second session included actual aural recordings from a loudspeaker. Our NN was able to accurately identify the input signals and told us something more had happened to the signals (i.e., the output values indicated that the signal had been modified).

The values for channels 1-3 indicated a different magnitude compared to the signal corresponding output values from the signal generator source. This magnitude distribution points out that the loudspeaker colored the input signal (see Figure 12). This coloration is typical of loudspeakers and gives each model its characteristic sound quality.

WHAT WAS ACCOMPLISHED

The task of building an artificial ear using NNs was successful. In fact, the results are beyond what we expected from such a simple model. Our experiments show that a NN is a very robust classifier since it can take noisy signals and extrapolate their underlying clean features.

We also showed that a NN can be trained on data from a signal generator and, using the same learned weights with minor retraining, it can recognize signals generated by a loudspeaker and microphone.

The results of these experiments are encouraging, especially when considering applying NNs to nonlab environments. The success of the artificial ear means that real-world signals can be recognized in a complex audio environment.

WHAT NOW

Using NeuralEar, the artificial ear program is fun and easy. You can train it to recognize any type of signal. Once the hardware and software are in place, you should be able to start a training session in about 5 minutes! Through NNs, you can use your intelligence to train a computer to solve your tricky problems. ❑

Dr. Samir Sayegh is a professor of Physics at Purdue and Indiana *University. He specializes in medical imaging, neural modeling, and neural net applications.*

Seibert Murphy, president and engineering director of Sound Sciences, specializes in acoustic signal processing, sound and vibration test system design, and transducer performance analysis. Current research in biological acoustics is aimed at developing a machine capable of high-speed acoustic imaging. He may be reached at murphys@cvax.ipfw.indiana.edu.

SOURCES

Data acquisition board
Deux Ex Machina Engineering, Inc.
1390 Carling Dr., Ste. 108
St. Paul, MN 55108
(612) 645-8088

NeuralEar System
Sound Sciences, Inc.
P.O. Box 9555
3182 Mallard Cove Ln., Ste. 002
Fort Wayne, IN 46899-9555
(219) 436-8705
Fax: (219) 436-8705

NeuralEar data acquisition system (executable control program, DynaMind 4.0 graphical NN software, and précis sigma-delta A/D converter board) $649
DynaMind 4.0 graphical NN software $195
DynaMind Developer Pro NN development system . $1795
précis sigma-delta A/D converter board $350

IRS

401 Very Useful
402 Moderately Useful
403 Not Useful

DESIGN CONTEST WINNER

FEATURE ARTICLE

Gregg Norris & Eric Wilson

T'ne Tye Mouse

An Ocular Prosthesis

The Eye Mouse offers people with extreme disabilities the opportunity to control a computer. An electrode interface enables a person to manipulate a common mouse pointer using his or her eyes.

For thousands of people, an extreme disability such as severe cerebral palsy or amyotrophic lateral sclerosis (ALS) deprives them of the use of their limbs and facial muscles. It is extremely difficult for them to express themselves through speech or bodily movement. Approximately 30,000 people are currently afflicted with ALS. Another 5,000 cases are reported each year in the U.S.

Cerebral palsy is more common. Every year, 1 in 1,000 infants is born with CP in the U.S. In many cases, diseases such as these damage a majority of the nervous and muscular systems in the body, but leave the brain and eye movement unimpaired.

In these cases, the person may rely on eye movement for communication. Intentional, electronically detected eye movements, interpreted on a computer, offer a rich medium for expression. Luckily, it is not too difficult to detect eye motion by analyzing the electrooculogram signal (EOG).

In this project, we constructed the Eye Mouse (EM), which detects changes in the EOG that result from looking up, down, left, or right. These changes in eye position correspond to cursor movements on a computer screen. The user can also select a screen item once the cursor has reached its target. Unlike the standard mouse operation which requires users to press and release mouse buttons to select and initiate actions, Eye Mouse uses eye blinks.

Some secondary design constraints were imposed to make the overall system more marketable:

- Ease of operation-no complicated sequences of eye movements
- Low cost-parts run about $150.00 (infrared eye devices can cost $1000s)
- Simple design-easily manufactured
- Compact4.7" x 2.4" x 1.6"
- Simple to power-one 9-V battery (draws -50 mA)
- Electrical safety-500-V isolation at 10-µA leakage current
- Compatibility-works with any PC with a Microsoft mouse driver

EM mouse movements can control commercially available programs for disabled persons such as HandiWord or HandiKey, which enable the user to write without a keyboard. Also, new software for the EM is fairly easy to write since the mouse interface is common and easy to use (especially with Windows). For example, an alphabet, displayed on screen, can be selected using a cursor that moves around and selects the letters forming sentences.

EOGBACKGROUND

Like the electrocardiogram (ECG) and the electroencephalogram (EEG), the electrooculogram (EOG) is classified as a bioelectric phenomenon. However, it is not as widely known because its clinical use is limited. The EOG encompasses all the changes in potential which emanate from the eye orbit during eye movement. This microvolt signal propagates through the extra cellular fluid in the head and is easily detected by scalp electrodes on the face.

It has been demonstrated that the front of the eye is more positively charged than the back of the eye due to the higher numbers of negatively charged neurons that make up the retina at the back of the eye. If a voltmeter is placed between the front of the eye and the back, it registers somewhere in the range of 100 µV.

The eyes, therefore, act like a dipole in space (if you remember your physics) and create an electrochemical field in the surrounding extra cellular fluid. With a look up, the region above the eyes becomes more positively charged than the region below the eyes. The opposite is true if the person looks down. If the person looks right, the region to the right of the eyes becomes more positive, and vice versa if the person looks left. In addition, with blinks, the eyes actually roll up for a split second while the lids are closed. This produces an EOG signal similar to looking up. These potential differences around the eyes can be transduced by scalp electrodes and detected by an instrumentation amplifier.

To measure EOGs or any other biopotential on the body, one cannot simply connect wires between the body and an amplifier. Although there are currents and potentials existing on the skin surface, they are created by ions, not electrons. To measure these signals with electronic equipment, an electrode interface must be used. The electrode is essentially a transducer, which converts ionic currents into the electronic currents necessary to drive an instrumentation amplifier.

The EM uses the most common type of electrode used for biopotential detection: the nonpolar silver-silver chloride (Ag-AgCl) electrode. An electrolyte chlorine gel aids ion conduction between the skin and the electrode. The gel must make good skin contact for the ionic currents to transfer to the electrode properly.

Before electrode placement, the skin must be rubbed with isopropyl alcohol to remove the dead skin and oil that impedes good electrode conduction. In fact, a good way to tell how well the electrode-skin interface has been prepared is to measure the DC resistance between two electrodes using a DMM. For our purposes, a resistance of 10-kΩ or below is enough to pick up a good EOG signal.

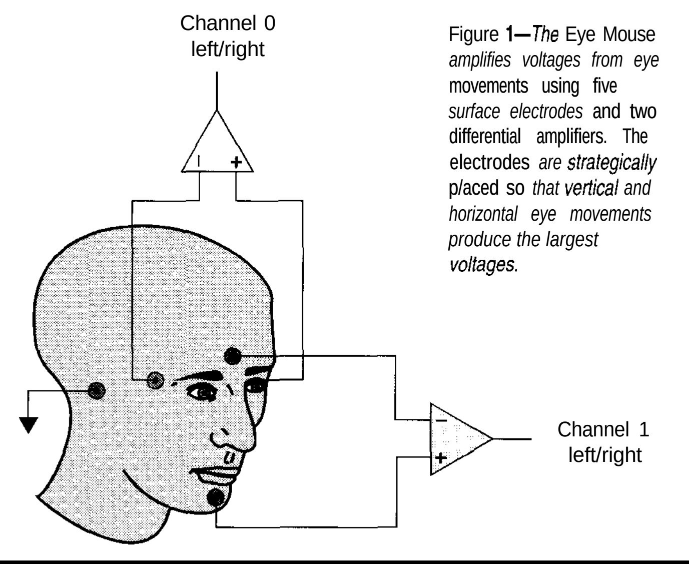


Figure 1—*The Eye Mouse amplifies voltages from eye movements using five surface electrodes and two differential amplifiers. The electrodes are strategically placed so that vertical and horizontal eye movements produce the largest voltages.*

INTERFACE HARDWARE

A Burr Brown INA102 instrumentation amp amplifies the EOG by 100. The INA102 output is proportional to the voltage difference between the positive and negative inputs. The electrode connected to the positive input is called the *positive* electrode and the one connected to the negative input is called the negative electrode. A separate amp detects the right or left and up or down eye movements.

Figure 1 illustrates electrode placement, and Figure 2 is a schematic of the Eye Mouse amplification. For right or left detection, the positive electrode is placed on the left temple and the negative electrode over the right temple. Looking to the left causes the positive electrode to be more positive than the negative electrode by about 100 µV, resulting in an output from the INA102 of about 10 mV. Looking to the right results in an INA102 output of about -10 mV.

To detect up or down movement, the positive electrode for the second INA102 is placed on the chin while the negative electrode is placed on the forehead. If the person looks up or down, the voltage changes on the electrodes are about three times less than when the eyes are moved left and right. In addition, a reference electrode, tied to signal ground of the INA102s, must be placed on the head. We chose to place it on the mastoid bone behind the right ear, a common placement for EEG recordings.

After the instrumentation-amplifier stage, a high-pass filter (3-dB

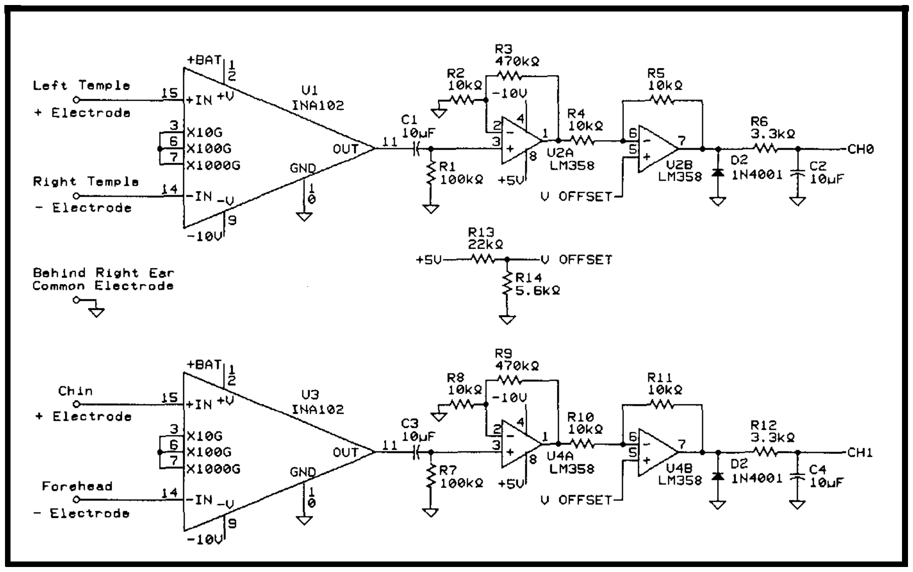


Figure 2—*The horizontal and vertical eye movements are amplified and filtered on separate channels before being read into the PIC16C71. The first stage of each channel is a differential amplifier with a gain of 100. Next, the DC and "electrode drift" component of each signal is removed before they are further amplified by 470. Finally, a 2.5-V offset is added to each signal before they are filtered to attenuate noise above 4.8 Hz.*

cutoff = 0.16 Hz) removes any DC artifact from the signal. This must be done to prevent saturation in the second amplifier stage. Because the electrodes in the bipolar pair are separated by some distance, a small DC scalp potential could exist between the two electrodes, even when the eyes are still and looking straight ahead. This occurs naturally from differences in skin thicknesses over the scalp and in ionic variations over the head due to sweat, electrode preparation, and variations in the electrode gel over time.

After the low-frequency component of the EOG is removed from both channels, the signal is amplified to

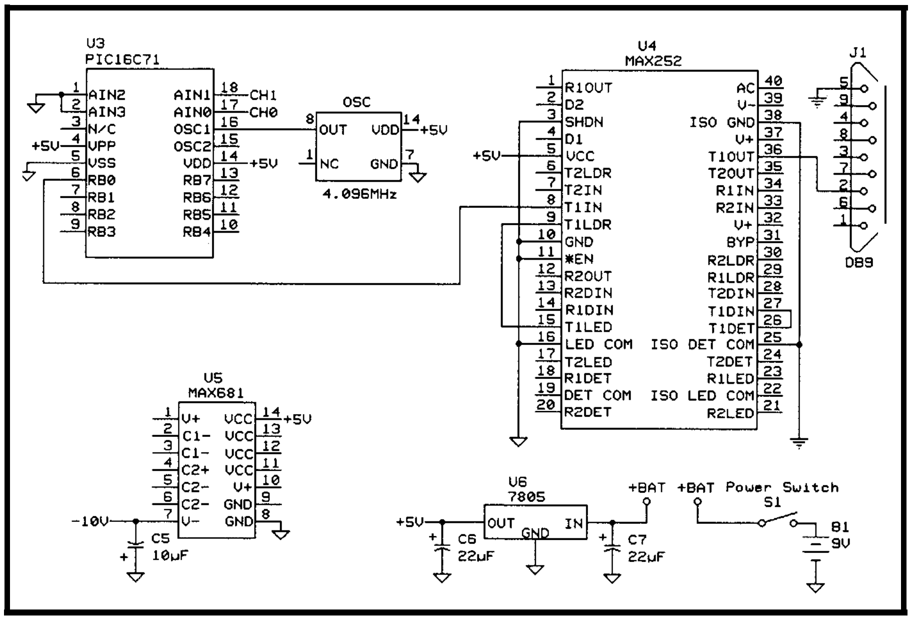


Figure 3—*The PIC16C71* interprets *the eye-movement voltages and outputs serial data that mimics a computer mouse* being moved in fhe *same* direction as the eyes *are moving. For* electrical safety, this serial data is passed through *an isolation* transceiver *before being sent to the computer mouse port. Two voltage regulators convert 9 V from a battery into +5 V and ±10 V for the rest of fhe circuit.*

± 1 .O V. After that last amplification stage, the signal is offset by 2.0 V since the PIC16C71 A/D converter must receive a signal in the O-5-V range. At this point, the signal level is normally at 2.0 V and can range from 0 V to 5 V, depending on where the eyes are looking.

The signal is then passed through an antialiasing filter (3-dB cutoff = 4.8 Hz). The filter, however, must not be made too low in frequency or it attenuates the signals from quick eye movements such as a blink. Finally, the filtered EOG is fed into the PIC1671.

THE MICROPROCESSOR, SERIAL INTERFACE, AND POWER

The PIC16C71's small size, low cost, design simplicity, and low power made it our microcontroller choice. Figure 3 is the circuit diagram for the PIC, power, and optically isolated serial interface.

EOG signals are fed into channels 0 and 1 of the PIC16C71. These are processed by the software, and the resulting serial mouse commands are sent to the MAX252 via RBO. The MAX252 isolation RS-232 transceiver output connects to the receive line of the PC serial port.

The whole circuit is powered by a single 9-V battery that is regulated to 5 V using a 7805 linear regulator and converted to -10 V using a MAX681 voltage-converter chip. The entire circuit draws about 50 mA of power, most of which (20 mA) is drawn by the RS-232 transceiver's internal optocouplers.

ELECTRICAL SAFETY

If the reference electrode behind the ear is connected to wall ground, the person would be badly shocked if they came in contact with the 120-V wall power.

To prevent this, the EM is electrically isolated from wall power and ground. The MAX252 transmits serial data through internal optocouplers so that none of the wires on the Eye Mouse side are in electrical contact with the wires in the serial cable. The chip only allows up to 10 µA of leakage current at up to 500 V.

SOFTWARE

The bulk of the software was designed by creating a flowchart that represents how the EOG data is analyzed and interpreted to create mouse commands. A scaled-down version is shown in Figure 4.

The software uses 30 of the 36 PIC16C71 general-purpose registers. Since most of the values are constants, register usage can be reduced. However, we chose to implement these values as variables to accommodate possible future code enhancements. We used 350 of the 1024 words of program memory.

The software is made up of three modules:

1. RTCC overflow ISR
2. main loop
3. serial communications

The RTCC overflow ISR in invoked every time the onboard real-time clock/counter overflows. The RTCC, which is programmed to overflow every 4 ms using the prescaler, simply increments two counters every 4 ms. The counters keep time in the main loop.

The main loop is programmed to run every 28 ms and can be broken into two parts:

1. reading and analyzing the EOG data
2. sending commands to the mouse driver running on the PC

Simply put, part 1 of the main loop determines what mouse commands are to be sent and communicates them to part 2 using a set of flags. Part 2 checks the flags and sends the appropriate commands.

The serial communications code is made up of two subroutines: send-out and bit_delay. send_out serially dumps out a 7-bit data word to port B, bit 0. bit_delay times the serial transfer. The Eye Mouse protocol mimics the Microsoft Mouse Protocol of 7 bits, no parity, and 2 stop bits at 1200 bps.

EYE MOUSE OPERATION

Removing the DC artifact from the EOG signal poses a special problem for control based on the EOG signal. Removing the near-DC component causes the filtered signal output to go to 0 in the steady state. Therefore, direct mapping of left-right and up-down EOG voltages to x and y cursor coordinates on the computer screen is confounded.

For example, if an operator stares at the upper-left corner of the screen, the cursor gradually drifts back to the center of the screen. Additionally, calibration is difficult. The EM avoids the problems associated with direct gaze mapping.

EM operation only requires that the person look in the desired direction for more than 0.5 s to move the cursor to that direction. After the cursor is moving, it continues to move in that direction until given a stop command, a command made with two eye blinks, less than 1 s apart. The operator is free to look anywhere during cursor movement.

To move the cursor to the left, the user looks to the left for more than 0.5 s and then look straight ahead at the computer screen. The cursor starts moving to the left at a predetermined speed. Once the cursor reaches the desired position on the screen, blinking twice stops the movement. Whenever the cursor is stopped with a double blink, the person selects an item underneath the cursor by merely blinking twice for a single click or blinking three times for a double click.

When we blink, our eyes move upwards; the eye lids are closed for a split second. To the EM, a blink appears as if the person looks up for a very brief time (about 0.25 s). To differentiate between blinking and looking up, a blink is defined as any up signal that lasts less than 0.5 s. To avoid registering a blink due to a noise, the up signal must last at least 0.1 s.

RESULTS

An example describing the communication protocol for the EM is shown in Figure 5. In this description, the signals necessary to move a cursor in a desired trajectory are shown. First, the cursor is moved up, then right, down, and left. At the end, the mouse button is clicked once, then twice.

Notice the slight amount of crosstalk between the two EOG channels. This is especially evident on the up and down channel since this channel had three times the sensitivity of the right-left channel.

When the person looked to the right (c), there was also a decrease in voltage on the up and down channel which crossed the lower threshold. There is also a small glitch on the up-down channel due to left eye movement at (g). On the right-left channel, small glitches occur whenever the person blinks, but their size is negligible.

The large glitches on the up-down channel are a potential problem. Since the glitches could cross the threshold,

WE'RE No. #1

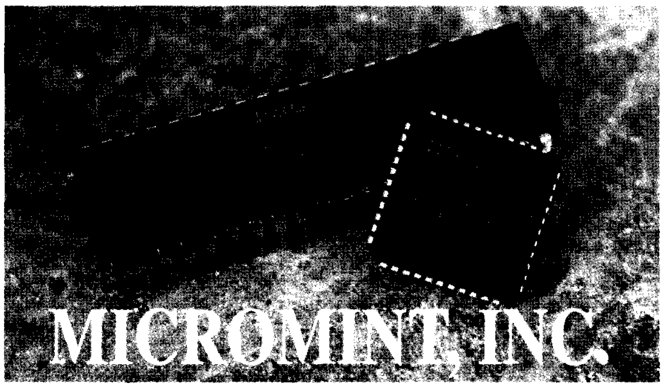
80C52-BASIC CHIPS IN VOLUME

Micromint's 80C52-BASIC chip is an upgraded replacement for the venerable Intel 8052AH-BASIC chip. Ours is designed for industrial use and operates over the entire industrial temperature range (-40°C to +85°C). Available in 40-pin DIP or PLCC.

80C52-BASIC chip $19.00

OEM 100 qty. **$12.00**

BASIC-52 prog. manual **$15.00**



Call (203) 871-6170 or 1-800-635-3355 to order

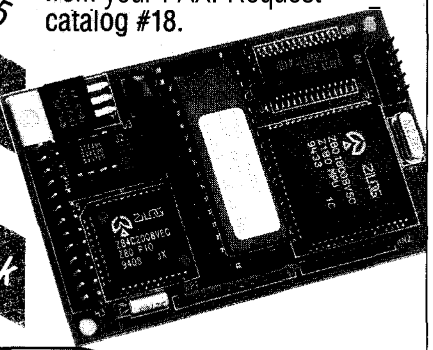
MICROMINT, INC. 4 PARK STREET, VERNON, CT 06066

#119

$79 QTY 1 **Micro Genius**™

Smaller than a credit card and C-programmable! Ask for our free Dynamic C™ demo disk, Call our AutoFAX 916-753-0618 from your FAX. Request catalog #18.

ADC
RS232/RS485
Flash EPROM
Power Supply
Real-Time Clock
Parallel I/O



Z WORLD ENGINEERING

1724 Picasso Ave.
Davis, CA 95616
916.757.3737
916.7535141 FAX

#108

they could be misinterpreted as up or down eye movement when the eyes were actually moving horizontally. To avoid this error, the software first checks the right-left channel for a threshold crossing. If one is not found, the software checks the up-down channel.

Therefore, when a person looks to the left or right, a horizontal move is performed. Any resulting glitch on the other channel is ignored. In fact, once moving, all voltages on both channels are ignored unless a distinct double blink occurs on the up-down channel to stop movement.

During the testing of the EM, the system performed well. However, the EM occasionally misinterpreted eye commands. These errors usually occurred because of noisy eye movement. For example, if the eyes are fixed straight ahead while the head is turned to the left, Eye Mouse reads a signal that looks like the eyes moving to the right. When this occurred, the person simply blinked twice to stop the errant movement.

A nice feature of the Eye Mouse is that while the cursor is moving, it ignores all eye movement except a double blink. Therefore, when traveling in a desired direction, it is nearly impossible to suddenly change direction due to noise.

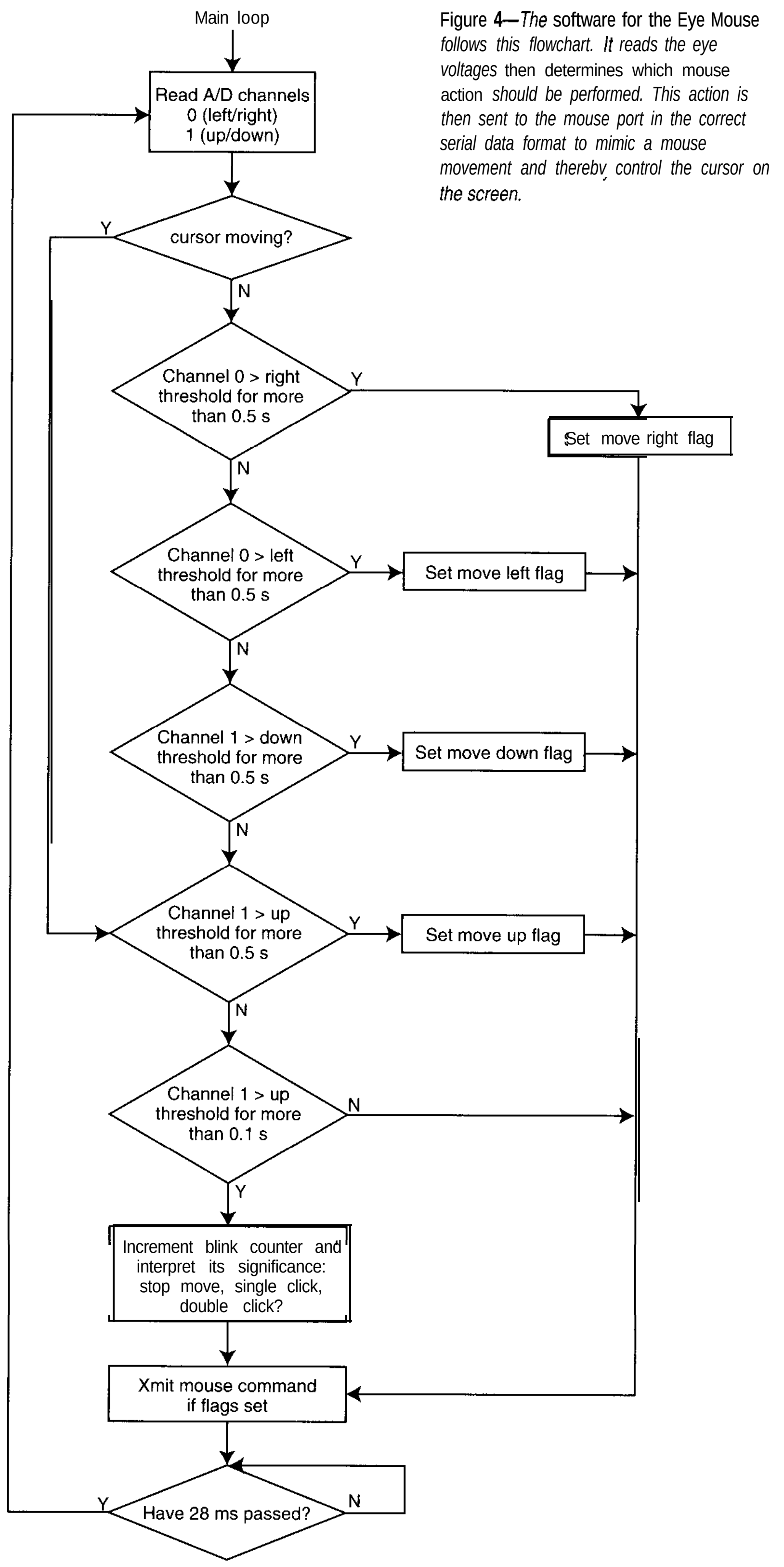
FUTURE IMPROVEMENTS

Some design enhancements are being made to facilitate system setup for the patient or therapist in the field. Because the strength of the EOG signal varies somewhat from person to person and from day to day, depending on electrode contact and placement, threshold adjustment can become a nuisance.

To solve this problem, Eye Mouse could analyze the change or differential in the EOG signal instead of using a level detection scheme. This system would be more robust and virtually immune to the slowly changing potentials from electrode drift since thresholds would no longer be a key factor.

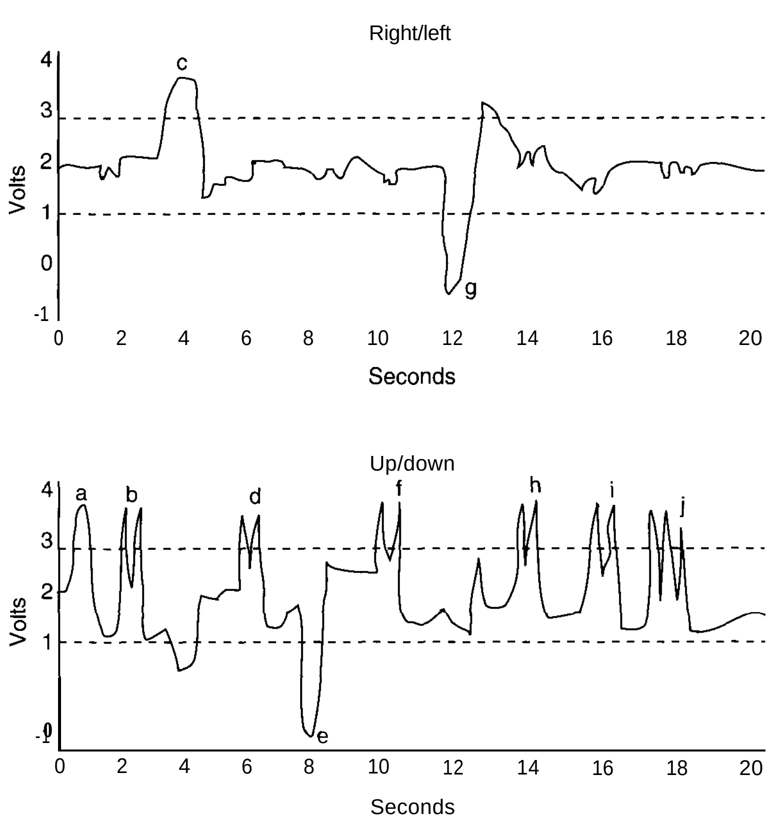
As well, for electrical safety, it might be prudent to use a plastic box or nonconductive box rather than the aluminum one made for this prototype. Whenever the chassis is connected to the serial cable, it becomes grounded from the cable's connector. Even though none of the circuitry comes in contact with the chassis, fluid could be spilled inside the box, defeating the built-in isolation.

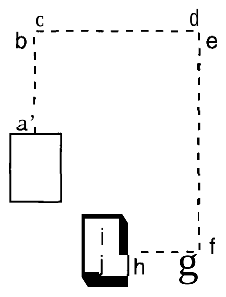
It would be even better to have the ability to transfer data without any



Main loop
Read A/D channels 0 (left/right) 1 (up/down)
cursor moving?
Y
N
Channel 0 > right threshold for more than 0.5 s
Set move right flag
Channel 0 > left threshold for more than 0.5 s
Set move left flag
Channel 1 > down threshold for more than 0.5 s
Set move down flag
Channel 1 > up threshold for more than 0.5 s
Set move up flag
Channel 1 > up threshold for more than 0.1 s
Increment blink counter and interpret its significance: stop move, single click, double click?
Xmit mouse command if flags set
Have 28 ms passed?

Figure 4—*The* software for the Eye Mouse *follows this flowchart. It reads the eye voltages* then determines which mouse action *should be performed. This action is then sent to the mouse port in the correct serial data format to mimic a mouse movement and thereby control the cursor on the screen.*





a = Start moving up
b = Stop moving
c = Stat-l moving right
d = Stop moving
e = Start moving down
f = Stop moving
g = Start moving left
h = Stop moving
i = Single mouse click
j = Double mouse click

Figure 5—*Here* are *the* resulting eye voltages *when the eyes move in the following pattern: look up, blink twice, look right,* blink twice, look down, *blink* twice, look *left,* blink twice, blink twice again, blink *three times.* This sequence moves *the* cursor up, stops *if, moves right, stops* if, moves down, *stop* if, moves *left,* stops if, and then performs a single and double mouse click.

don't like windows?

don't care for r&d?

sorry.

Then we just can't help you. But if you're looking for a high-capacity, user-friendly EDA system, we've got just what you need. Say "hi" to EDWin, your new companion in Electronics Design. EDWin features seamless integration between modules, so you can finally kiss the tedious concept of front- and back annotation goodbye. EDWin gives you all the tools you'll need, and is so user-friendly you can even compile your own custom toolboxes. So easy to learn, you'll be up and running in minutes, EDWin also features nice pricing, starting at just $495.
Make your appointment with us today for the EDWin evaluation package. Welcome.

Vision EDA Corp.
995 E Baseline Rd. Ste 2166,
Tempe, Arizona 85283-l 336
Phone: l-800-EDA-4-YOU, or (602) 730 8900
Fax: (602) 730 8927

ELECTRONICS DESIGN FOR WINDOWS

EDWIN

EDWin is a trademark of Norlinvest Ltd. Windows is a trademark of Microsoft Corp.

#1

conductive material between the EM and PC. This could be achieved with fiber optic or infrared light link between the EM and the PC.

CONCLUSION

As a communication aid designed for the severely disabled, Eye Mouse allows the user to mimic the actions of an ordinary PC mouse with the movements of their eyes. In a sense, it acts as an ocular prosthesis for persons unable to communicate through speech or body movement.

Since the EM emulates the actions of a mouse, it is easy to imagine it controlling mouse-driven software. It could provide specialized word processing capabilities or help control an environment by turning on the TV, calling a nurse, and so on.

In fact, the EM could greatly enhance commercially available software packages for severely disabled people. Such packages currently respond to the actions of simple mechanical devices such as pressure switches.

Finally, with all the improvements in mind, the parts cost of the Eye Mouse would still be amazingly cheap (about $180.00). The system would still be affordable to any disabled person in the hospital, home, or at work. ❑

Gregg *Norris and Eric Wilson are electrical engineers who have been designing and building embedded systems for five years. Gregg's background in robotics and Eric's in biomedical applications prepared them for their current work with Varian Ion Implant Systems, a semiconductor equipment manufacturer. They may be reached at* gregg.norris@varian.com *and* eric.wilson@varian.com.

SOURCES

INA102 Instrumentation Amplifier
Burr Brown Corporation
P.O. Box 11400
Tucson, AZ 85734
(520) 746-1111

MAX252 Isolation Transceiver and MAX681 5-V to ±10-V Converter
Maxim Integrated Products
120 San Gabriel Dr.
Sunnyvale, CA 94086-9892
(800) 998-8800 (free samples)
(408) 737-7600
Fax: (408) 737-7194

Electrode Cable E449 5 Lead Snap Array
In Vivo Metric
P.O. Box 249
Healdsburg, CA 95448
(707) 433-4819

Electrodes A7 Adult Disposable Lead
Lok, Inc.
500 Airport Way
Sandpoint, ID 83864
(208) 263-5071
Fax: (208) 263-9654

IRS

404 Very Useful
405 Moderately Useful
406 Not Useful

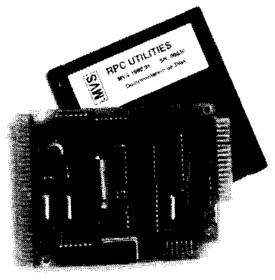
REMOTE POWER CARD!

3 VERSIONS:

RESET
WATCHDOG RESETS PC IF IT HANG3 FOR HARDWARE OR SOFTWARE REASONS

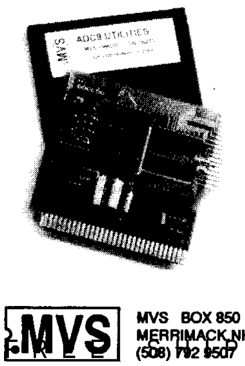
PHONE
TURN ON PC WITH PHONE, SHARE VOICE / MODEM LINE, CONTROL AC APPLIANCES

TIMER
WAKEUP OR SHUTDOWN PC, LATE NITE BACKUP / MODEM, CONTROL AC APPLIANCES



**95$** 27$ OEM

ALL MVS CARDS INCLUDE ASM, BASIC, AND C SOURCE FOR PC OR SBC



8 CHAN ADC
DATA AQUISITION, SERVO CTL, AUDIO
8-BIT RESOLUTION 22KHZ SAMPLE RATE
SHARP CUTOFF ANTI-ALIAS FILTER
CREATE STEREO BLASTER (.VOC) FILES
95$

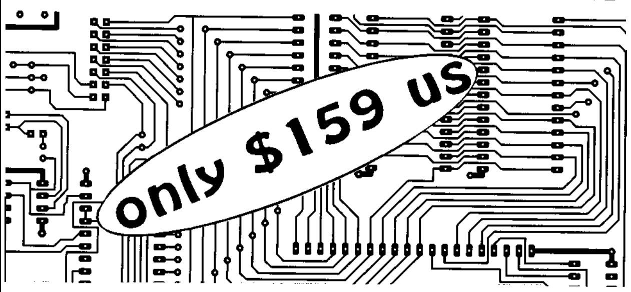
2 CHAN DAC
VOICE MAIL, MUSIC, ALARMS, CTL VOLT
8-BIT RESOLUTION 44KHZ SAMPLE RATE
PLAYS MONO / STEREO BLASTER FILES
FUNCTIONS AS DIGITAL ATTENUATOR TOO
75$

MVS
MVS BOX 850
MERRIMACK, NH
(508) 792 9507
5 YEAR LIMITED WARRANTY
FREE SHIPPING IN USA

#110

CADPAC

PCB AND SCHEMATIC TOOLS
2 for the price of 1
PCB II & SUPERSKETCH FOR NEW LOW LOW PRICE



** features comparable to packages costing thousands!
** must be tried to be believed
**"easiest product to use for designing PCBs"
** customers call it "the 8th wonder of the world!"

!! REDUCED !!

R4 SYSTEMS INC.
1100 GORHAM ST. SUITE 11 E-332
NEWMARKET ONTARIO
CANADA L3Y 7V1
905 898-0665 FAX 905 898-0683

FREE DEMO WRITE OR CALL TODAY

BBS 905 898-0508 (96,8,1,N)

FEATURE ARTICLE

Don Lancaster

The Quest for Magic Sine Waves

Upping Power Electronics Efficiency

A little magic goes a long way in replacing PWM. Don creates his magic through canceling harmonics. As he points out, the tricky part is not coding the binary sequences, but searching for the right 420-bit words.

There is a lot of fresh interest in higher-power digital sine waves. People want to know about everything from induction motor-speed control to electric autos, UPS power quality, phone ringers, and off-grid solar inverters.

As Figure 1 shows, the key to all of these applications is to start with a DC supply and some switches. You then digitally flip these switches in some sequence to try to produce a clean power sine wave in your motor or transformer winding.

This switching arrangement is usually known as an ***H-bridge drive.*** With switches in positions B and D or in A and C, there is zero new motor or transformer current. When in positions A and D, ***positive*** motor current gets added to the present waveform. But, in positions B and C, ***negative*** current is added (i.e., current is removed).

Obviously, we need to provide a variable frequency and amplitude with low harmonics and a zero DC term. To maximize efficiency, we want to use as few switch flips as possible per cycle. Plus, of course, we want to end up purely digital and microcontroller friendly. The central quest: find some magic new digital-sine-wave switching scheme that meets these goals.

The old-line stock solution here was once known as..

PULSE-WIDTH MODULATION

. .or PWM for short. With PWM, you start by using a high-frequency carrier-say a 20-kHz carrier for 60-Hz power. As with most FM or PM schemes, the duty cycle varies. By averaging or integrating the carrier's duty cycle, a low-frequency modulation can be recovered. This integration normally gets done when the inductance of the motor winding acts as a low-pass filter.

The big attraction to PWM is the ease with which both amplitude and frequency can be changed. An analog PWM is easily generated using a sawtooth and a comparator.

But, there are a few grievous flaws to PWM. With PWM, there are inherently a lot of switch flips per cycle. Each flip costs you dearly with high-frequency losses. Losses mean higher temperatures, more expensive drive transistors, and larger heatsinks.

PWM carrier amplitude is ***always*** larger than the fundamental.

Worse, each transition in stock PWM is typically a ***double*** flip. You change ***both*** sides of your bridge at once, which gives you an additional two times efficiency penalty. The number of switch flips for each cycle usually is totally independent of your output amplitude, so low amplitudes leads to lousy efficiency. There is always substantial energy kicking around at unwanted high frequencies, which can lead to whine or noise.

While all of the low harmonics can theoretically be eliminated, the real world may not work that way. Any noise, distortion, quantization, nonhnearities, or a DC term in your PWM modulation directly shows up as output imperfections.

USING "MAGIC" SINE WAVES

Lately, I've explored a new magic sine wave approach that seems to offer many advantages over the stock PWM-not to mention being utterly fascinating and highly addictive.

Take a big long string of ones and zeros. When repeated, this string possesses a Fourier series consisting of a fundamental and some harmonics. By picking all of your ones and zeros precisely right, we can force most of the lower harmonics to zero and *still* provide a variable amplitude output—just like PWM.

This ploy is very microcontroller friendly. For amplitude 78, you look up sequence 78 in a table and then send it. For frequency, you adjust the dwell time between bits.

There is no high-frequency PWM carrier. All those switch flips can be dramatically minimized. And, there's no modulation or integration hassles. Besides, no noise or distortion-prone analog sawtooth is involved.

So far, I have found that the most interesting results use word lengths of 210 and 420 bits. Addiction comes when you try to find useful methods to work around exhaustively searching for all of the possible 420-bit words.

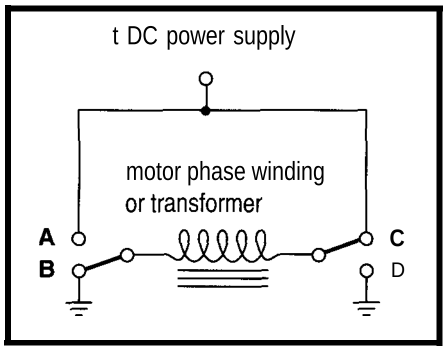


Figure I--Typical/y, power *sinewaves are created from a* DC supply by *flipping switches in the proper sequence to synthesize a sine-wave current of fhe* desired *amplitude,* phasing, *and* frequency. *In* position A-D, a *positive current is added. In position B-C, a negative* current is added *(i.e.,* current is removed). Positions B-D *or A-C neither add nor remove any current.*

SHORTER SOLUTIONS

Let us start with the simpler and shorter sequences of Figure 2.

My personal preference here is to use the general-purpose PostScript computer language because it's incredibly friendly, intuitive, and serendipitous. As well, it's freely available as a GhostScript clone.

I've written a simple PostScript FOURIER.PS analyzer. This has been posted both to my own GEnie Post-Script Round Table (PSRT) and the Circuit Cellar BBS. It analyzes any sequential string of ones and zeros and gives you the size of the fundamental and any interesting harmonics. This data is given in several formats with or without selective oversampling.

For most any sequence length, we can guarantee no DC term by having an equal number of +1 and -1 switch states. This is often important to eliminate any level bias or iron-saturation effects.

We can force all even harmonics to zero by providing for half-wave symmetry. Since your second harmonic is by far your most serious problem, you usually force zero evens.

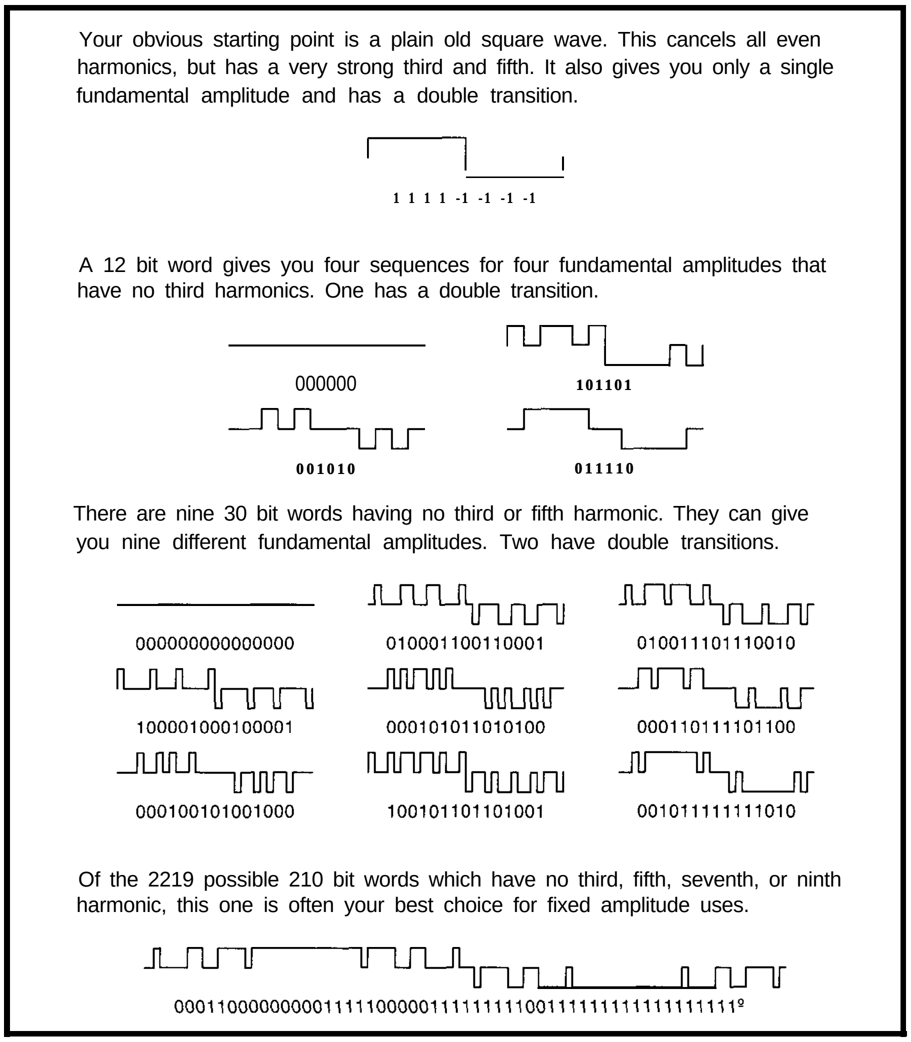
For half-wave symmetry, you mirror top to bottom, but *not* left to right. For instance, on a 30-bit word, when bit *n* is a zero, bit *n*+15 must also be a zero. If bit *m* is a +1, then bit *m*+15 must be a –1.

Your obvious starting point is a square wave. However, there are three major gotchas. The first and worst is that there is only one nonzero fundamental amplitude available. A second is that the third harmonic is a horrendous 33% and the fifth is a largish 20% of the fundamental. Finally, since you are always flipping from +1 to -1, you have to include an efficiency-robbing double transition.

Still, a square wave is cheap and simple. When you can live with fixed amplitude and if strong harmonics are no problem, go for it.

Solutions that are trinary and permit values of +1, -1, and 0 appear to lead us to cleaner and more desirable results. If we never have any +1 right beside a -1, then *all* double-switch flips can be avoided.

Figure 2 also shows us four 12-bit waveforms which completely cancel their third harmonics. Because of the equal number of +1 and -1 states, there



Your obvious starting point is a plain old square wave. This cancels all even harmonics, but has a very strong third and fifth. It also gives you only a single fundamental amplitude and has a double transition.

1 1 1 1 -1 -1 -1 -1

A 12 bit word gives you four sequences for four fundamental amplitudes that have no third harmonics. One has a double transition.

000000 101101
001010 011110

There are nine 30 bit words having no third or fifth harmonic. They can give you nine different fundamental amplitudes. Two have double transitions.

000000000000000 010001100110001 010011101110010
100001000100001 000101011010100 000110111101100
000100101001000 100101101101001 001011111111010

Of the 2219 possible 210 bit words which have no third, fifth, seventh, or ninth harmonic, this one is often your best choice for fixed amplitude uses.

0001100000000001111100000111111111100111111111111111111º

Figure 2—*These* examples explore low-harmonic binary sequences starting *with a plain old square wave and finishing with a 2 10-bit word.*

Figure 3—*These 104 magic power sinewaves are selected for maximum efficiency by allowing only 12 or fewer quadrant transitions. Harmonics 0, 2, 3, 4, 5, 6, 8, 9, 10, 12, 14, 15, and 16 are 0. Harmonic 7 is 0 or low. Only the first quadrant of each 420-bit word is shown.*

```
000 - 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
001 - 000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000
002 - 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
003 - 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000000000000
004 - 000000000000000000000000000000000000010000000000000000000000000000000000000000000000000100000000000000000
005 - 000000000000000000000000000000000000000001000000000000000000000001000000000000000000000100000000000000000
006 - 000000000000000000000000000000000100000000000000000000000000000000000000001000000000000000000000000010000
007 - 000000000000000000000000000001000000000000000000000100000000000000000000000000010000000000000000100000000
008 - 000000000000000000000000000000000100000000000000000000010000000000010000000000000001000000000001000000000
009 - 000000000000000000000000000100000000000000000001000000000000000000000000100000000000000100000000000000100
010 - 000110000000000000000000000000000000000000000110000000000000000000000000100000000000010000010000000100000
011 - 000000000000010000000000000000000000000000010000000000110000000000000000001100000000000000000011000000000
012 - 000000000000000000000000000000001100000000000000000001100000000000000000000000000001110000000000100000000
013 - 000000000000000000000010000000000000000010000000000010000000010000000010000000100000001000000000000000011
014 - 000000000000000000001000000000000000001000000000000010000000001000000001000000010000001000001000001000000
015 - 000000000000000000001000000000000000010000000000010000000010000000010000000100000001000000001000000000010
016 - 000000001100000000000000000000000000000000111000000000000000000000000000000001100000000011100000000000000
017 - 000000000000001000000000000000000000000001110000000000000000000000000011100000000000001000000000001110000
018 - 000000000000000000000000001100000000000000000000001110000000000000000000011100000000001000000001110000000
019 - 001111000000000000000000000000000000000000000111100000000000000000000000000001111000000000000001100000000
020 - 111000000000000000000010000000000000000001110000000000000000000000000001111000000000000000001111000000000
021 - 000000000000000000000000000011000000000011000000000000000000000000001000111100000000000000000000011110000
022 - 000000110000000000000000000000000000000011100000000000000000000000000111110000000000000000000000001111000
023 - 000000000011000000000000000000000000011000000000011100000000000000000001110000110000000000000111100000
024 - 000000000000000000000001100000000000000001100000000000110000000000000000000000111111000000000000110000000
025 - 111100000000000000000010000000000000000000111100000000000000000000000100001111000000000000000001111000000
026 - 000011100000000000000000000000000000000000001111000011000000000000000000000011110000000001100001111000000
027 - 011000000000000000000000110000000000000001100000000000001110000000000000110000000001110000000000111000000
028 - 000000110000000000000000000000000000111000000000000000000001110000000000000110000000011100000000011100000
029 - 011111000000000000000000000000000000000000011111000000000000000000000111110000000000000000000000000011110
030 - 000000000000000000000000110000000000000001110000000000000000000000011111100000000000000000011111000000000
031 - 000000000110000000000000001000000000001100000000011111000000000000000000001111110000000000000001111100000
032 - 000000000100000000000000000111000000000000000000000001111110000000000000000000000111111100000000110000000
033 - 000000000000000001000000001000000000000000000001111100000000000000000000000001111111000000000000001110000
034 - 000001100000000000000000000011001110000000000000000000111100000000000011111100000000000100000001111100000
035 - 000010000000000000000011000000000000000111001000000000000000000001111000000000000111111000000000000011111
036 - 000000000000000000000010000001100000000011100000000000000000000001111111100000000000000000000111111000000
037 - 001111000000000100000000000000000000000011111100000000000000000000011111111100000000000000000100111111000
038 - 111111000000000000000100000000000000000000000111111000000000000000000111111100000000010000001111111000000
039 - 011111000000000000001000000000000000000001111110000000000000000100000011111110000000000000111111110000000
040 - 000110000000000000000000111000000000000000001111100000000000000001000011111110000000000000001111111000000
041 - 000000000000001100000000000000000011110000000000000001111110000000000000111111100000000000111111110000000
042 - 000000000000000001100000000001100000000011000000000011111110000000000000000001111111110000000000000001111
043 - 000000000000000000000001110000000000000000000001111100000000000000000000000001111111110000000000111100000
044 - 001110000000000000000110000000000000000011111100000000000000011000111111111000000000000001111111110000000
045 - 000000000000000001000000001110000000000000000000011111100000000000000001111111000000000000000111111110000
046 - 011110000000000000001100000000000000000000001111111000000000000000001111111000000000000011111111110000000
047 - 000001000000000000000011100000000000000111100011100000000000000000111111111000000000000001111111100000000
048 - 000000000000000001000001110000000000000011000111110000000000000000111111111000000000000000111111111000000
049 - 000000110000000000000000000000111000000111000000000000000000011111111100000000000000000011111111111000000
050 - 000000000000000011100000000001000000000000000001111111110000000000000000001111111111100000000000000111110
051 - 000000000000000011000000110000000000000011001111110000000000000001111111110000000000001111111111110000000
052 - 000001100000000000000000000000011110000110000000000000000000111111111000000000000000000111111111111000000
053 - 000000000000000011100000000001000000000000000011011111110000000000000000000011111111111100000000000111110
054 - 001100000000000000011100000000000000000000011111111000000000000000001111111110000000000111111111111000000
055 - 000000110000000000000000011110001000000000000001111111110000000000000111001111111110000000111111111100000
056 - 000000000000000011100000000000000000000000000111111111100000000000000000111111111111100000000000011111101
057 - 000000000000000001110000001000000000000000111011111000000000000000011111111110000000001111111111111000000
058 - 000111100000000000000000000001100000111100000000000111111100000000000111111111110000000001111111111110000
059 - 000000000000000001111000000000000000000000000111111111000000000000000000011111111111110000000000011111110
060 - 000000000000000001110001000000000000000001111101110000000000000000111111111111100000000011111111111100000
061 - 000001100000000000000000000000001111110000000000000000000000111111111111110000000000000111111111111110000
062 - 000000010000000000111000000000000000000000011111111110000000000000000011111111111111100000000000001111111
063 - 000000000000000001110000000000000000000001111111110000000000000000111111111111100000000000111011111111000
064 - 000011000000000000111000000000000000001111111110000000000000011100111111111111000000000111111111111110000
065 - 001111100000000000000000000011111000001111000000000011111110000000001111111111111000000000011111111111000
066 - 001111110000000000000000000000111000000111100000000011111111100000000011111111111111000000000011111111111
067 - 000001100000000000000111000000000000000000000111111111100000000000001100111111111111110000000001111111110
068 - 001111111000000000000000000011000000111110000001111111100000000000111111111111111000000000111111111111000
069 - 000000000000000001111000000000000000001000111111111100000000000000001111111111111111100000000111111111100
070 - 111111000000000000011000000000000001111111110000000011111100000000111111111111110000000011111111111110001
071 - 011111000000000000000000000001111100000011100000000011111111100000000111111111111111000000011111111111100
072 - 011111000000000000011100000000000000001111111111000000000111111000000111111111111111110000000001111111111
073 - 001111000000000000001110000000000000000011111111110000000000111111000001111111111111000000011111111111100
074 - 011111110000000000000000000111110000000111110000001111111110000000000111111111111111000000111111111111100
075 - 000000000000000001110000000111000000000111111000111000000011111111111110000000111111111111000000111111110
076 - 011111111000000000000000000111000000000111110000111111111000000001111111111111100000001111111111111111000
077 - 011100000000000000000111100000000000000011111111100000000000001111111000111111111111111111000011111111100
078 - 000000000000000000111100000000011100000000011111100001110000000001111111111111111100000000111111111111110
079 - 111100000000000000001111000000000000111111111000000000001111111110000111111111111000000011111111111111100
080 - 111111000000000000000000000000111111000000001110000000111111111100000001111111111111100000001111111111111110
081 - 000000000000000000011100000000111100000000000011110001110000000011111111111111111111000000011111111111111
082 - 001100000000000000000111100000000000000000111111111000000000111111111111100000001111111111111111111111100
083 - 001000000000000000001111000000000000001111111100000000011111111111000000111111111111110001111111111111100
084 - 000000000000000001111000000000001110000111110000000001111111111111000000111111111111111000011111111111110
085 - 001100000000000000111100000000000000011111111100000001111111111110000001111111111111111111000011111111111
086 - 000001111000000000000000000011110000001000000111111111111110000000001111111111111110011111111111111111100
087 - 011111100000000000000000000111111100000000111000001111111111100000001111111111111110001111111111111111100
088 - 000000000000000000011100000011111100000000001111000111111000000011111111111111111111111000011111111111111
089 - 111000000000000001111000000000000000001111111111100000000111111111100011111111111111000001111111111111110
090 - 011000000000000001111000000000000000011111111111100000001111111111000111111111111111000111111111111111110
091 - 000000000000000000000110000011111110000000000001110011111111100001111111111111111110011111111111111111110
092 - 000001100000000000000000111111111000000000000001111111111111000001111111111111000011111111111111111111110
093 - 111110000000000000110000000111111000000001111000011111111111110000011111111111110001111111111111111111110
094 - 001111100000000000000000000111111110000000000011100111111111111110000011111111111111111111100111111111111
095 - 000000000010000011110000000000000000111111111111000001111011111111110000111111111111111111100111111111111
096 - 000000000000000111110000000000000000111111111110000001111111111110000111111111111111111111111111111111110
097 - 000000000000000000001100011111111100000000011001111111110001111111111111111100011111111111111111111111110
098 - 000000000011000011100000000000000011111111111110000100001111111111111111111110001111111111111111111111111
099 - 000000010000000000000011111111101000000000011111111111111000111111111111111110011111111111111111111111111
100 - 000011100000000000000000001111111110000000001111111111111011100011111111111111110011111111111111111111111
101 - 111000000000000000111100000000110000111000111111000011111111111111100011111111111111101111111111111111111
102 - 000111100000000000000000111111111000000000011111111111111111100011111111111111111111111111111111111111111
103 - 000000000111110000011000000000000001111111111111001111100000111111111111111111111111111111111111111111111
```

is no DC term. And, because of half-wave symmetry, there are no even harmonics present.

The fundamental amplitudes of the four waveforms are 0, 57, 80, and 100%. On the strongest amplitude, the fifth harmonic is a tolerable 20%, and the seventh is at 14%. Clearly, we have a much better solution here.

By using exhaustive searches or dumb luck, we can find sequences of ones and zeros which have low (but nonzero) values for any particular harmonic. These can sometimes be useful. But, to force a truly zero *n*th harmonic, you have to go to bit lengths that are a product of *n* and some other numbers.

For instance, the only bit lengths that offer hope of completely canceling out a third harmonic are 3, 6, 9, 12, 15.... The only lengths, which completely cancel a fifth are 5, 10, 15, 20, 25.... Thus, by exclusion, the only bit lengths that completely cancel *both* third and fifth are 15, 30, 45, 60....

Figure 2 shows nine 30-bit trinary waveforms that have no DC term, no even harmonics, a zero third, and a zero fifth. The strongest fundamental peak is the supply voltage at 1.05, which has an 11.8% seventh and a zero ninth. You can pick amplitudes of 0, 27, 43, 53, 65, 70, 80, 87, and 105%.

In many cases, the output power is more of a concern than the output voltage. These same 10 waveforms give you relative powers of 0, 7, 17, 25, 38, 44, 58, 69, and 100%.

We now have a way to adjust *both* the frequency and amplitude of our magic sine wave. But, the total number of useful 30-bit amplitude steps is sorely limited. To beat this, we go to longer bit lengths.

A reasonable goal is to try to find 100 amplitudes in 1% steps. This set meets most motor-control needs without going to insanely long words or uselessly high frequencies. In Figure 2, you finally go to one super magic 210-bit waveform, which is the product of 2, 3, 5, and 7. So, certain carefully selected sequences should cancel out. This particular wave shape has a zero DC, zero evens, and zero third, fifth, seventh, fifteenth, and twenty-first. Your eleventh is an amazingly low 1.08%. The thirteenth ain't half bad either at 8.9%.

True, lots of gruesome higher harmonics exist. After all, the square corners in your wave shape do have to come from somewhere-a fundamental and a pitifully weak eleventh certainly can't hack it by themselves. But, very high harmonics are usually easy to filter-none of them is worse than the third on a plain old square wave. There's a mere seven transitions per quarter cycle for a total of 28!

It is interesting to compare this waveform against a 210-bit PWM wave shape. PWM might require 420 double transitions compared to 28 single transitions for the magic 2 10 sine wave. While I can't claim that my magic sine wave is 30 times more efficient, I can say that the transition losses are often 30 times worse with PWM and I can get by with significantly smaller heatsinks and drivers.

I previously selected a hundred useful 210-bit solutions from the 2219 or so possible having zero third, fifth, or seventh. These have been posted as **HACK87. PDF** and related files. But, I was not happy with the excessive number of transitions on some selections, their amplitude uniformity, or their distortions. Thus, our current magic-sine-wave quest here is to find some better 420-bit words.

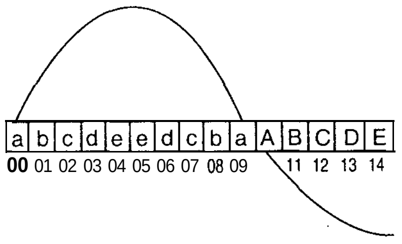
NEEDLES IN HAYSTACKS

The big problem with longer bit-length words is that there are great heaping bunches of them. Even a 60-bit word has 1,152,921,504,606,846, 976 states. So, neither an exhaustive search nor any random or Monte Carlo selections suffice.

Instead, we have to work smarter and not harder. We have already noted that half-wave symmetry gets rid of even harmonics. It also slashes the total number of cases by chopping the analyzed bits in half.

Forcing Odd Harmonics to Zero

Say you want to find 60-bit words having a zero third harmonic. Use quarter-wave symmetry to guarantee no even harmonics and no cosine terms. Draw one-quarter-cycle worth of third harmonic sine and label the bits as shown here.



Now a is the contribution by bit #00, determined by the sine of the angles at the start and end of the bit. And, *A* is minus this value. (It does not matter what the value of *a* is.) The only way you get a perfect cancellation is to ensure that a, A, and *A* cancel each other.

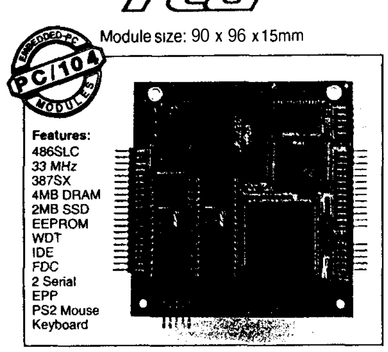
This can only happen if all three bits are zero or if there is only one a present to cancel out the value of *A*. For full cancellation, you can write these equations:

$$00 + 09 = 10$$
$$01 + 08 = 11$$
$$02 + 07 = 12$$
$$03 + 06 = 14$$
$$04 + 05 = 15$$

Note further that only ones and zeros are acceptable in these equations. Since 00 and 09 cannot simultaneously be one, we need only consider three cases for each equation.

Out of the 32,768 possible quarter-cycle words, there are only 3^5 or 243 quarter symmetric 60-bit sequences having a zero third harmonic.

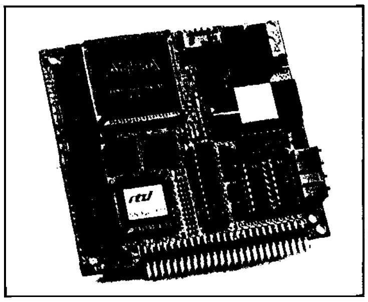
REDUCE THE STACK!
Use fully integrated PC/104 CPU and DAS modules from RTD®



PC/104 Compliant PC-AT SBC
CMi486SLC-1: $597 /100 pieces
2MB DRAM & SSD software included.

6 PC/XT/AT **Single Board cpuModules™:**
- 486DX, 486SLC. 386SX, F8680 & V30 CPUs
- 80387SX math coprocessor socket on-board
- 2MB, 4MB or 8MB DRAM installed
- Two 32-pin SSD sockets & support software
- IDE, floppy & CGA controllers
- RS-232/422/485 serial ports
- Bidirectional parallel, keyboard & speaker ports
- Keypad scanning & virtual device support
- Power management & single +5V supply

7 utilityModules™:
- SVGA, Ethernet & intelligent GPS modules
- PCMCIA carriers for Types I, II & III cards



PC/104 Compliant 200 kHz Analog I/O Module
DM5408-2: $498 /100 pieces

17 DAS dataModules®:
- 12 & 14-bit ND conversion up to 200 kHz
- Random scan, burst & multiburst
- Pre. post & about triggers
- 1K channel-gain scan memory with skip bit
- 1024 sample A/D buffer with data marker
- 12-bit analog outputs
- Bit programmable digital I/O with Advanced Digital Interrupt modes
- Incremental encoder interfaces
- 4-20 mA current loop source
- opto-22 compatibility
- Low power & single +5V power supply

For technical *specifications and data sheets on PC/104, ISA bus and Eurocard products, call*
RTD USA Technical FaxBack®: 1 (814) 235-1260
RTD USA BBS: 1 (814) 234-9427

RTD® Real Time Devices USA
200 Innovation Boulevard • P.O. Box 906
State College, PA 16804-0906 USA
Tel: 1 (814) 234-8087 • Fax: 1 (814) 234-5218
RTD Europa RTD Scandinavia
Fax: (36) 1 212-0260 Fax: (358) 0 346-4539

RTD is a founder of the PC/104 Consortium and the world's leading supplier of PC/104 CPU and DAS modules.

#112

To perform a traditional Fourier analysis, find out how much of the wave shape can be absorbed by a harmonic sine and cosine term. To minimize work, it is often useful to force all your cosine terms to zero. To do this requires quarter-wave symmetry, in which the left and right sides of each half waveform mirror each other.

Thus, by using sine terms only, you again cut the number of bits in half at the risk of losing certain solutions. For a 420-bit result, you only need to analyze 105 bits.

This still may take a while-even with PostScript. We need to dramatically reduce the candidate patterns.

One trick is shown in the sidebar on forcing odd harmonics to zero. To cancel out the third harmonic, certain bit combinations must add up to zero. These can lead to a series of linear equations. To cancel the fifth, other combinations must add to zero (this is also true for a seventh). Solving these equations together greatly reduces your search problem.

The sidebar uses a 60-bit word as an example. There is a 15-bit quarter word and 32,768 states if we try an exhaustive search.

Can we further reduce this?

The third harmonic has three quarter cycles in the quarter word of bits *abcdeedcbaABCDE* with $-a = A$ and so on. Now, a is an angle with some sine-it makes no difference what its value is. Since all sines end up different from each other, the only way you can get perfect cancellation is if $a + a = A$. This example says that bit 00 plus bit 09 must equal bit 10. Bit position 01 plus bit position 08 must equal bit position 11 and so on. Otherwise, it won't cancel.

This gives us five equations to cancel out the third. Using 1¼ cycles per quarter word, call the fifth harmonic bits *abccbaABCCBAabc*. Next, write out three equations that perfectly cancel your fifth. Now you have eight equations and fifteen unknowns. Apply substitution to reduce this to one equation with seven unknowns, which drops us from 32,768 states to a mere 128!

Wait! There's more. On a quarter cycle, these equations are binaries, whose only valid values are zero or one. So, if a binary equation of $U + V + W + X = Y + Z$ shows up, you do *not* need to check all 64 cases. Go through the possibilities which allow only ones and zeros for any value, and you end up with only 15 valid solutions. You can safely ignore the others. Similarly, $J + K = L$ allows only three, not four, valid solutions.

By restricting your arithmetic to binary values, you can further reduce the 128 states. Even then, additional testing may still be needed.

At 60 bits, 31 different amplitudes have zero third and zero fifth. Of these, 010001110111111 is the "best" quarter-cycle example at 1.02 amplitude and a 3.4% seventh. The ninth is zero.

Even if you allow "weak" values of third and fifth, you still only get 48 or so total different amplitudes at 60 bits. But, not all of those are useful owing to uneven spacing or an occasional high seventh.

As you increase the total bits in a word, the selection of useful solutions goes up at an infuriatingly slow rate. This leads us to the need to explore a lot of very long sequences.

In the case of 420 bits, we have a 105-bit quarter word. We can write 35 equations with no third, 21 with no fifth, and 15 with no seventh, or 71 equations with 105 unknowns. This reduces to something like 1 equation with 34 unknowns. Actually, it's somewhat worse than this because of certain obscure cancellations. But, many are still left.

By using binary-only solutions, the number of combinations can be further pared. The harmonic contribution of any chosen bit is a fixed numeric value. These values can be placed in a look-up table eliminating on-the-fly trig or slow multiplication.

Restricting the number of transitions or ones in the words are two of the final reduction tools.

SOME RESULTS

A lot of juggling is still involved to come up with useful results. You want as many amplitudes as you can get that are evenly spaced with as few transitions as possible and low distortion. Not surprisingly, you often have

NEW!

A Complete Family of Data Acquisition Products from

American Eagle Technology

- Data acquisition rates from 100kHz to 1 MHz.
- Newest designs incorporate the latest technology: FIFO buffers, dual-DMA, REP INSW data transfers, programmable gains, etc.
- Simultaneous sampling option for all boards.
- Lowest prices on the market for comparable performance.
- Digital I/O w/ on-board relays.
- 16-bit analog output (D/A).
- Large inventory of popular items for next day delivery.
- Also available: signal conditioning, multi-port serial boards, device programmers, logic analyzers, & much more.

Only American Eagle Technology Gives You All This Free Software:

- *WaveView* menu-driven software
- *Discstream* high-speed streaming software
- Complete software developers' kits for both DOS and Windows
- Drivers for all popular application programs such as LabView, LabWindows, NOTEBOOK, SnapMaster and DASyLab

American AET Eagle Technology

526 Durham Rd
Madison, CT 06443

Call: (203) 245-6133
Fax: (203) 245-6233

Send for Your FREE Catalog Today!

#113

Figure 4—*These 104 magic power sine waves are selected for minimum total harmonic distortion with all odd harmonics present. Only the first quadrant of each 420-bit word is shown.*

```
000 - 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
001 - 000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000
002 - 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000000000000
003 - 000000000000000000000000000000000000000000000000100000000000000000000000000000000000000100000000000000000
004 - 000000000000000000000000000000000000010000000000000000000000000001000000000000000000000100000000000000000
005 - 000000000000000000000000000000000000000001000000000000000000000000100000000000000000000100000000000000000
006 - 000000000000000000000000000000000010000000000000000000000000100000000000000100000000000000000000000010000
007 - 000000000000000000000000000100000000000000000000000100000000000000000001000000000000000000000000100000000
008 - 000000000000000000000000000000010000000000000000000100000000000000010000000000000100000000000001000000000
009 - 000000000000000000000000000100000000000000000000100000000000000100000000100000000000000100000000000000100
010 - 000000000000000000000001000000000000000000010000000000001000000000001000000000001000000000100000000100000
011 - 000000000000000000000000001000000000000000001000000000000100000000000010000000001000000000100000000010000000
012 - 000000000000000000000001000000000000000001000000000000010000000000100000000010000000001000000000100000000010
013 - 000000000000000000000100000000000000001000000000000100000000010000000010000000010000000100000010000010000
014 - 000000000000000000000010000000000000001000000000000100000000010000000010000000100000010000001000001000000
015 - 000000000000000000000010000000000000001000000000001000000000100000000100000001000000010000000010000000010
016 - 000000000000000000001000000000000000100000000000100000000100000001000000010000001000000100000010000001000
017 - 000000000000000000000100000000000000010000000000010000000010000000100000001000000100000100000100000100000
018 - 000000000000000000001000000000000001000000000001000000000100000001000000100000010000010000010000010000010
019 - 000000000000000000010000000000000100000000010000000100000010000001000000100000100000100001000001000000100
021 - 000000000000000000100000000000000100000000010000000100000001000001000001000001000010000100001000100010000
022 - 000000000000000001000000000000001000000000100000001000000010000010000010000010000100001000100001000010
023 - 000000000000000001000000000000010000000001000000010000010000010000100001000010000100001000010000100
024 - 0000000000000000100000000000001000000000100000001000001000001000001000010000100001000100001000100001000
025 - 0000000000000001000000000000100000000100000001000000100000100001000010000100001000010000100010000100001
026 - 0000000000000001000000000001000000000100000010000010000100001000010000100010000100010000100010001000100
027 - 0000000000000001000000000001000000000100000010000010000100000100010001000100010001000100010001000
028 - 000000000000000100000000000010000000100000010000001000001000010000100001000100001000100010001000010001000100010001
029 - 000000000000000100000000000100000001000001000001000010001000010001000100100010001000100100
030 - 000000000000000100000000001000000010000001000010000100001000100010001001000100010010001001000
031 - 0000000000000010000000000001000000100000010000100001000010001000100010001000100010001001000100010010001
032 - 00000000000000100000000000100000010000010000100010001000100010001001000100100010010010010
033 - 000000000000001000000000001000000010000010000100001000100010001000100100010010001001001001000100100
034 - 00000000000000100000000001000000010000010000100001000100010001000100100010010010001001001001001001
035 - 000000000000010000000000100000010000010001000010001001000100100010010010010001001001001001001001010010010
036 - 00000000000001000000000100000010000010000100010001001000100100010010010010001010010010010010010100
037 - 00000000000001000000000100000010000010000100010001000100010010010001001001001001001001001001001
038 - 00000000000001000000000100000100000100010001000100010010001001001001001001001001001001010010010
039 - 0000000000000100000000010000001000010000100010001000100100010010010010010010010100100100101001010100100
040 - 0000000000001000000000100000010000100010001000100100100010010010010010010010100100100101001001
041 - 0000000000001000000000100000100001000100001001000100100100100100101001001010010010100101001010010
042 - 000000000000100000000010000001000010001000100010010001001001001001001001010010010100101001010010100
043 - 00000000000010000000010000001000010001000100010010010010010010010100100101001010010100101001
044 - 0000000000001000000001000001000010000100010010001001001001001010010010100101001010010101001010
045 - 000000000000100000000100000100010000100100010010001010010010100100101001010010101001010100101010100
046 - 0000000000001000000001000001000010001001000100100100100101001001010010101001010010101001010100101
047 - 00000000000100000000100000100001000100100010010010010100101001010010101001010100101010010101010
048 - 0000000000001000000010000001000100010001001001000101001001010010100101010010101001010101010010
049 - 00000000000100000000100000100010001000100100100100100101001010010101001010100101010101010010101
050 - 000000000001000000001000001000100001001000100100100100101001010010101001010101010100101010101010101010
051 - 00000000000100000000100000100010001001000100100100101001010010100101010010101010101001010101010101010
052 - 0000000000010000000010000010001000100100010010010100100101001010100101010101010010101010101010101
053 - 00000000000100000000100000100010000100100010010010100101001010010101010101001010101101010101010101010
054 - 00000000000100000000100000100010001001001001001010010010101001010101010101010101101010101010101010
055 - 000000000001000000001000001000100010010010010010100101001010101001010101010101010101010101010101
056 - 0000000001000000010000010001001001000101001001010010101001010101010101010101011010101010101010110
057 - 0000000000100000001000010001000100100100100101001010010101010010101010101010101011010101011010101010
058 - 00000000010000000100000100010001001001001010010100101010101010101010101010101101010101010110101
059 - 00000000010000000100001000010001001001001001001010010100101010101010101010101010101101010101101010110
060 - 000000000100000001000001000100010010010010100101001010101001010101010110101011010101101010110101011010
061 - 00000000010000000100001000010010001001001010010100101010101010101010101010110101010101101010110101101
062 - 0000000001000000010000100010001001001001001001010010101010101010101010101011010101101011010110
063 - 00000000010000000100001000100010001001001010010010101001010101010101010110101011010110101101011011010
064 - 000000000100000001000001000100010010010010100101010010101010101010110101010110101101010110110101101
065 - 00000000010000001000010000100100010100100101010010101010101010110101010110101101011010110101101101
066 - 00000000010000001000010000100100010010010100101001010101010101010101101010101101011010110110101101101101
067 - 0000000001000000100001000100010010010010100101010101010101010101101010110101101011011010110110110
068 - 00000000010000001000010001001000100101001001010101010101010101011010110101101011011011011010101101
069 - 00000000010000000100001000100010010010100101010010101010101011010101101011010110110110110110110110
070 - 000000000100000001000010001001001001001010010101010101010101101010110101101101101101011011011011011
071 - 00000000010000001000010001001001010010101010101010101010110101101011011010110110111011011011011101
072 - 00000000010000001000010000100100100101001010100101010110101010110101101011011011011101101101101110110
073 - 0000000001000000100001000100010010010100101010010101010101101010110101101101101101101101101110110110111011
074 - 000000000100000010000100010010010010100101010101010101011010110101101101101101101101110110111011101
075 - 00000000010000001000010001001000101001001010101010101010101011010110101101101101101110110111011101110110
076 - 0000000001000000100001000100100100101010010101010101101010110110101101101101110110111011011101101111011
077 - 0000000001000000100010001000100101001010010101010101010110101101011011011101101110111011101110111101
078 - 00000000010000001000010010001001001010010101010101010101101011010110110110111011011101110111011101110
079 - 0000000001000000100010001001001001001010100101010101101010110101101101101110110111011101110111011111011
080 - 00000000010000010000100010010010010100101010101010101101011010110110110111011011101110111011101111101
081 - 000000000100000100001000100100100101010101010101010101101101011011011011101101110111011101110111011110
082 - 00000000010000010000100010010010101001010101010110101101011011011011101110111011101110111101110111
083 - 0000000010000001000010010001001010010101010101010101101011011011011011101110111011101111011101110111101
084 - 000000000100000100001000100100101001010100101011010101101011011011011101110111011110111101111011110111110
085 - 00000000100000010000100100100101001010101010101101011010110111011011101110111011110111101111101111011
086 - 0000000010000001000100010010010101010010110101011010110110111011011101110111101111011110111110111111011
087 - 00000000100000010001000100100101001010101010110101101011011011101101110111011101111011110111110111110
088 - 0000000010000001000100010010100101010101010101101011011011101101110111011110111101111011111101111101111
089 - 00000000100000010001000100101001010010101010110101011011011011101101111011101111011110111110111111011
090 - 000000001000000100010001001001010010101010101011010110101101110110111011110111101111011111101111111110
091 - 0000000010000010000100100010100101010010101101010110110110111011101110111101111011111101111111011111
092 - 00000000100000100001001001001010101010101011010110110110111011101110111101111011111101111111110111
093 - 000000001000001000010010010010101010101010101101011011011101110111011110111101111111011111111101111111110
094 - 0000000010000010000100100101001010101010110101101101101110111011110111101111110111111110111111
095 - 000000001000001000100010010010101001010101011010110101101101110111011110111110111111011111111111101111
096 - 0000000010000010001000100101001010101010101101011011101101110111101111101111111101111111110111111111
097 - 00000000100000100010001001001010101001101010101101101101110111011110111110111111110111111111111101111111
098 - 0000000010000010001000100101001010101010101101011011011101110111101111101111111110111111111111111
099 - 0000000010000010001001001001010010101010110101101101101110111011111011111101111111111101111111111111111
100 - 000000001000001000100100100100101010101010110101101011101101110111101111101111111110111111111111111111
101 - 00000000100000100010010010010101010101010101101101101110111011110111111101111111111111111111111111
102 - 000000001000001000100100100101010010110101011011011011101110111101111101111111111111110111111111111111
103 - 0000000010000010001001001001010101010101011011011011101110111101111111110111111111111111111111111111
```

many near misses picking the best from a sorry lot.

At any rate, my selection for 104 efficient magic sine waves that have low numbers of transitions appears in Figure 3. Harmonics 0, 2–10, 12, 14–16, 18, 20–22, 24–28, and 30 are zero for most of the sine waves shown. The eleventh and thirteenth are often well under 10%, even before filtering!

An occasional entry needing a seventh of less than 1% has been thrown in where it significantly reduced the transitions. And, we had to borrow a few very low amplitudes from Figure 4.

Amplitude spacing is monotonic and in roughly 1% steps. There is significant step-to-step jitter owing to the random nature of the harmonic amplitude math.

The clocking rate for a 60-Hz output is 25.2 kHz. But your actual transition frequencies are much less than this. For all but one value, the maximum number of transitions is 12 per quarter cycle or 48 total.

ANOTHER APPROACH

An interesting alternative is shown in Figure 4. This gives you 104 magic sine waves having low distortion values, but has more transitions per cycle and poorer efficiency. Despite its weaknesses, it's still significantly better and simpler than PWM.

All the odd harmonics here are present, but weak—typically, in the 0.5% range at 46 or more decibels down. These values may be more useful when strong harmonic filtering is not a great option (i.e., when an induction motor control runs over a wide speed range).

A sneaky method is used to find these values. The trick is to make each bit contribute as much as possible to the desired fundamental. You need to pick a desired fundamental amplitude. Then, simply start at the middle of the quarter cycle, picking out only those bits that take the biggest bite out of the remaining amplitude, but never exceed it. This technique gives you amazingly good results—very fast, too.

I've purposely shown all of these results in binary form. Viewing your actual sine waves gives you insights into what is happening. Note particularly the efficient clumping of Figure 3 and the progressive build of Figure 4 and you can see efficiency versus distortion tradeoffs.

If you select more compact notation, be particularly careful that any leading zeros used in hex notation do not end up as portions of your actual output waveforms.

The general-purpose PostScript language handles 420-bit words with aplomb. The key is to put your ones and zeros into a long string. You can then manipulate the string.

IMPROVING LOW AMPLITUDES

The results of either method are better for higher amplitudes. There are simply not enough ones to be placed in useful enough positions to give us really superb low-amplitude results, especially under 15% or so. For some uses, lots of low values do not matter because one quarter amplitude is only 7% of total power. One tenth amplitude is only 1% power.

If lots of low amplitudes end up essential, it seems best to combine the magic sine waves with a second stepped-power selection scheme. You could maybe use half- versus full-wave rectification to chop down the input voltage by two or four, switched taps, or something similar.

Other options: forego quarter-cycle symmetry or try alternating amplitude states.

FOR MORE INFORMATION

To use these values, just stash them in a table and look them up as needed. Serial EEPROMs are a good choice in a PIC environment.

You'll find a lot of tradeoffs in Figure 3, so consider it a "director's cut." By allowing 15 or 16 transitions per quarter cycle, you can get more steps with better spacing, but at the price of lower efficiency. You can also permit some additional low-harmonic distortion to pick up new candidate values or select fewer amplitudes to improve both efficiency and distortion.

For multiphase motors, some magic sequences can be suitably delayed. Everything shown is shiftable by thirds for three-phase power-control applications.

Many thanks to math genius Jim Fitzsimons for his gracious help on this project, especially for thinking out all of the hairy parts.

Don Lancaster is the author of 32 books and countless articles. In addition to offering his own books, reprints, and various services, Don offers a no-charge technical helpline at (520) 428-4073. He may also be reached at synergetics@genie.geis.com.

SOFTWARE

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

IRS

407 Very Useful
408 Moderately Useful
409 Not Useful

The only 8051/52 BASIC compiler that is

100 % BASIC 52 Compatible

and

has full floating point, integer, byte & bit variables.

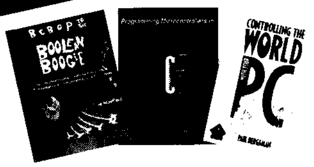
- *Memory mapped variables*
- *In-line assembly language option*
- *Compile time switch to select 8051/8031 or 8052/8032 CPUs*
- *Compatible with any RAM or ROM memory mapping*
- *Runs up to 50 times faster than the MCS BASIC-52 interpreter.*
- *Includes Binary Technology's SXA51 cross-assembler & hex file manip.util.*
- *Extensive documentation*
- *Tutorial included*
- *Runs on IBM-PC/XT or compatibile*
- *Compatible with all 8051 variants*
- ***BXC51 $ 295.***

508-369-9556
FAX 508-369-9549

Binary Technology, Inc.
P.O. Box 541 • Carlisle, MA 01741

#114

Program Microcontrollers! Control the World! . . . or just Bebop to the Boolean Boogie!



Programming Microcontrollers in C
by Ted Van Sickle, Motorola

Microcontrollers are showing up everywhere! But to use them, you have to program them, and that is a lot different from programming a PC. Van Sickle takes you inside today's hottest microcontrollers and shows you how to use C to program them for such tasks as motor control and digital signal processing. $29.95 and worth every cent!

Controlling the World with Your PC
by Paul Bergsman

Use your PC to control stepper and servo motors, monitor fluid levels, switch appliances on and off, monitor alarm systems and much more. All circuits connect directlytothe parallel printer port. Complete source code listings in BASIC, C, and Pascal are included on a 3.5-in floppy Each circuit is fully described with theory, parts values, and schematic. Only $29.95, including the disk.

Bebop to the Boolean Boogie
An Unconventional Guide to Electronics Fundamentals, Components, and Processes
by Clive "Max" Maxfield

The only book we know of where you can simultaneously learn about finite state machines, Reed-Müller logic, musical socks, and the best time of day to eat smoked fish! This reference is huge, with hundreds of diagrams that show you the Intimate details of integrated circuits, hybrids, multichip modules, and circuit boards Electronics will never be the same. $35.00, and worth the price for the glossary alone.

To order, call **1-800-508-BOOK**
Or send check/money order to HighText (include $4.00 S/H), CA residents include sales tax

HighText publications inc
P.O. Box 1489, Solana Reach CA 9207

#115

The Solution's in the CAN—Part 2

FEATURE ARTICLE

Brad Hunting

Putting CAN Online with a Multinode, Multiplatform Implementation

Determined to see if CAN really can, Brad puts CAN through all the paces—a multinode, multiplatform network, a PC, an 80C31, and an 80C592. It seems there ain't much this CAN cannot do.

Last month, I introduced the Controller Area Network (CAN) communication protocol (INK 58). This month, I'll put some hardware together to see if CAN really can.

For those who missed last month's introduction, CAN defines a small-area network protocol similar in scope to Bitbus or Arcnet. Although similar in scope, CAN's implementation differs significantly. CAN specifies a multidrop bus topology with broadcast message passing and carrier-sense, multiple access with collision detection. Any time any node senses the bus is free, it may initiate a message, and all nodes receive that message.

Added features include lossless arbitration and receive-message masking. CAN's collision arbitration relies on message priorities that result in higher-priority messages being transmitted and lower-priority messages halted when there is a collision. Since every message has a priority and it is illegal for two messages to have the same priority, one message always wins arbitration and is transmitted without loss of information or time.

Since CAN relies on broadcast message passing and every node receives every message, CAN controllers implement receive filters to prevent certain messages from being recognized. This prevents the CAN controller from interrupting the host processor when uninteresting messages are received.

This article describes three CAN network nodes. The first is a CAN controller running on an IBM-compatible PC, the second a CAN controller connected to an 80C31, and finally an 80C592, which is an 80C31 variant with an on-chip CAN controller.

THE SYSTEM

The system is shown conceptually in Figure 1 and as implemented in Photo 1. This three-node network has the three most likely implementations of a network node:

- a PC-compatible with an ISA bus-based communications controller

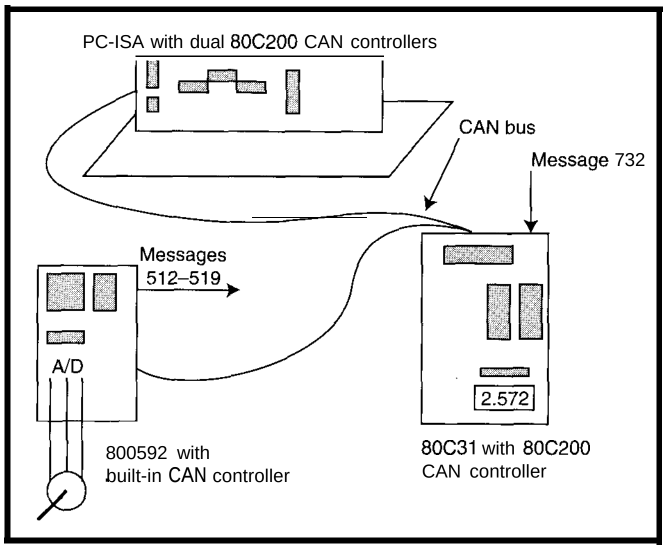

PC-ISA with dual 80C200 CAN controllers
CAN bus
Message 732
Messages 512–519
A/D
2.572
800592 with built-in CAN controller
80C31 with 80C200 CAN controller

Figure l--The CAN *network described here includes three different node* implementations, a PC/ISA, *a microcontroller, and an enhanced microcontroller.*

- a microcontroller with a separate communications controller
- a microcontroller with onchip integrated communications controller

The PC and the microcontroller are both using a Philips PCA80C200 stand-alone CAN communications controller for network interface. As mentioned, the Philips 80C592 has an onboard CAN controller plus a variety of other nice onboard peripherals, such as 8-channel IO-bit ADC, dual PWM outputs, three 16-bit timer/counters, a watchdog timer, and more.

SOFTWARE

The battle over C versus assembler still rages, but in this case using C proves quite beneficial. The software for all three hardware implementations has the configuration shown in Figure 2. All of the message handling, reading, writing, and polling is contained in one file-CAN I0.C. All three architectures access identical code to read and write messages.

The only code that differs between the three implementations is four lines

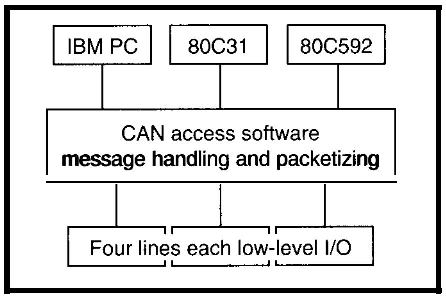

IBM PC
80C31
80C592
CAN access software
message handling and packetizing
Four lines each low-level I/O

Figure 2—*Message handling software for the three architectures is over 90% reused.*

of C in each. These four distinct lines are two lines to access the CAN controller hardware for a read, and two lines to access for a write. All of the communications code is reused for the three different architectures, except for four lines. Try getting the same level of code reuse in assembler!

In this demonstration code, the communication routines are not interrupt driven. If the code were changed to add interrupt handlers, it is likely that more of the code would differ and perhaps a little assembler would be required. The four lines of code that differ start with #if def to handle multiple architectures. To compile the code to execute under a specific architecture (PC, 803 1, or 80C592), define MS-DOS, F8031, or F80C592 during the compile. All of the software described in this article is available on the Circuit Cellar BBS.

The PC communication board described here has dual CAN controllers on a single board. One of the routines provided is an interrupt-driven routine to exercise the dual communications capabilities. Messages are passed back and forth between the controllers and the controller registers, and message packets can be printed to the screen.

PC software is also provided to monitor and display any messages passed on the CAN bus. Since CAN is a broadcast protocol, the PC simply receives all messages and prints them to the screen. Software for the 80C592 samples each of the eight onboard ADC channels and transmits the sampled values as messages 512–519.

Another routine runs on the 80C3 1 which specifically watches for message 732. When message 732 is received, the data carried in the packet

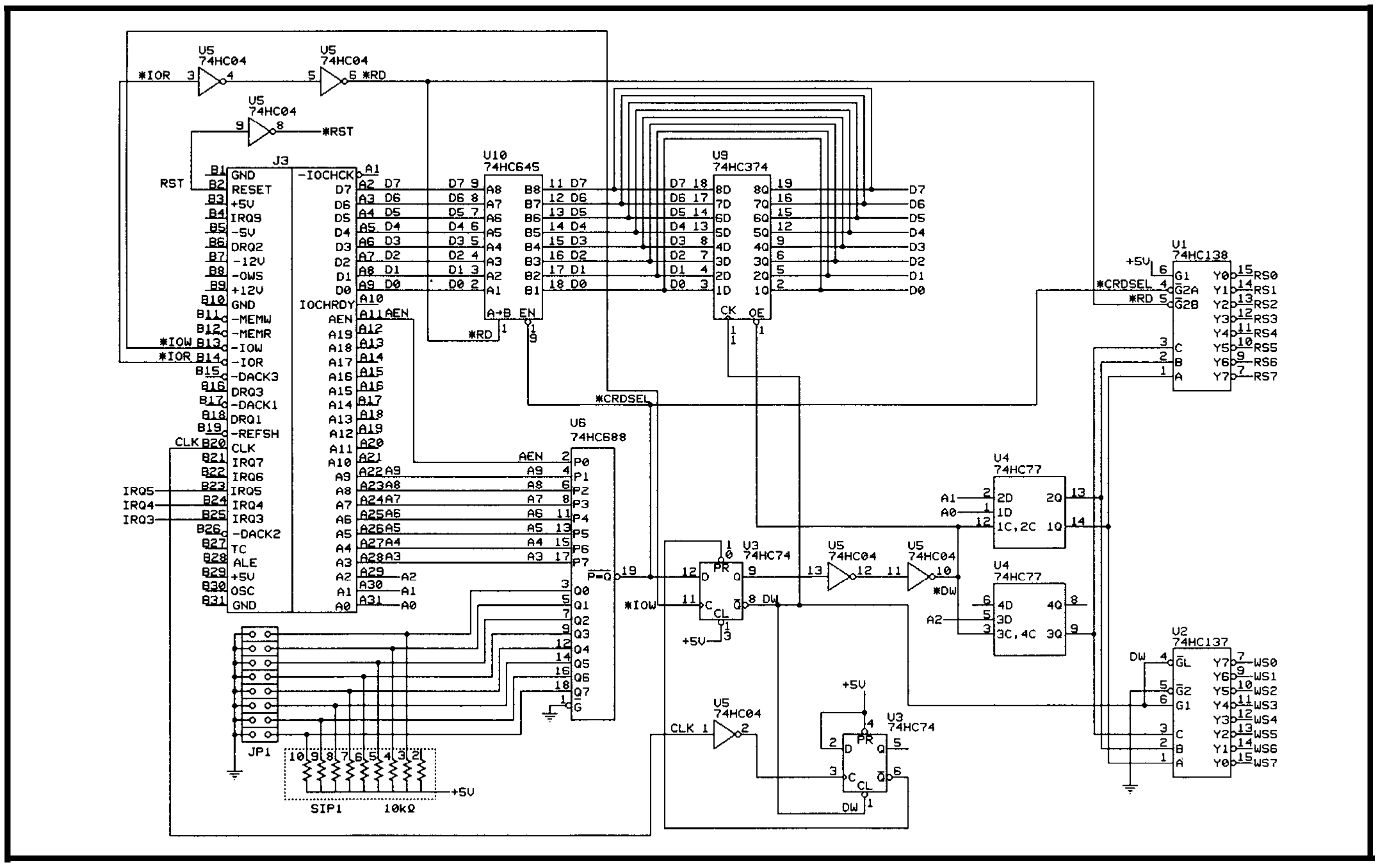
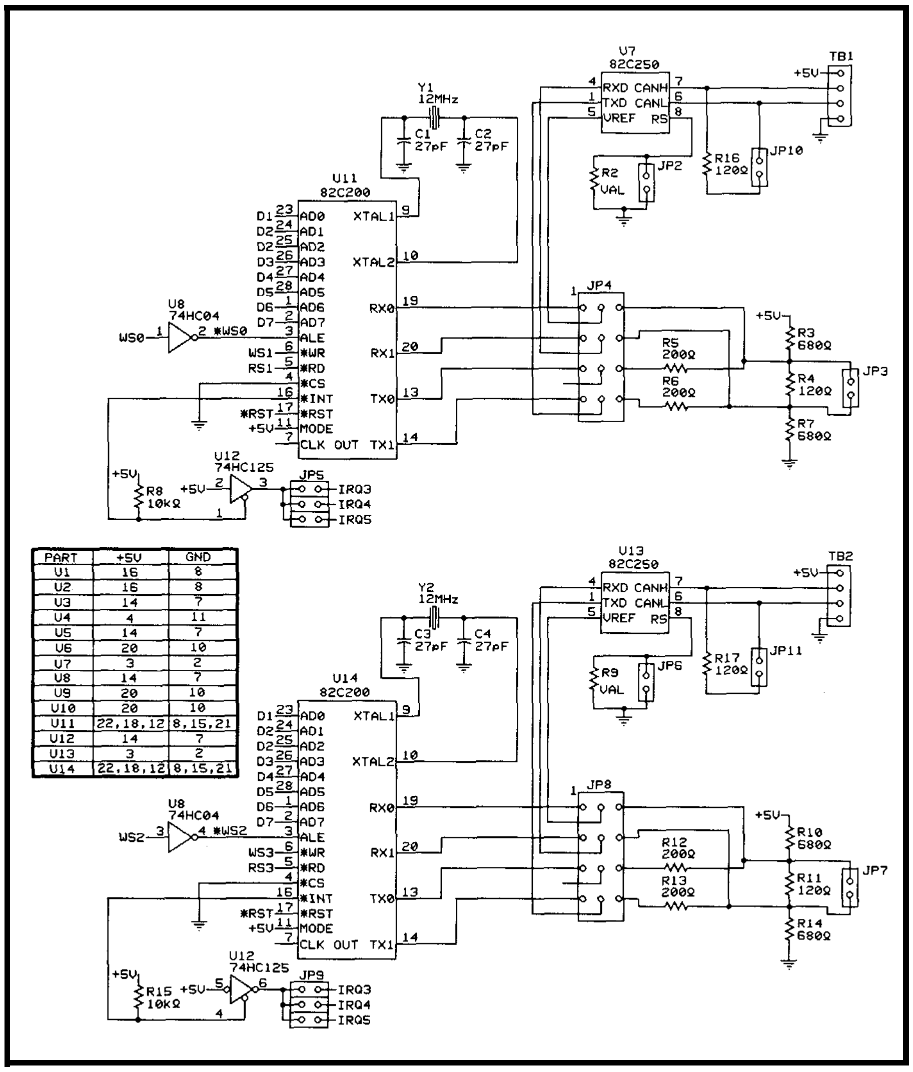


Figure 3a—*The PC bus interface is not critical for this application. Shown here is just another example of how to get on and off the PC bus.*



PART	+5V	GND
U1	16	8
U2	16	8
U3	14	7
U4	4	11
U5	14	7
U6	20	10
U7	3	2
U8	14	7
U9	20	10
U10	20	10
U11	22,18,12	8,15,21
U12	14	7
U13	3	2
U14	22,18,12	8,15,21

Figure 3b—*Interface to the CAN bus is accomplished with the Philips PCA80C200 CAN controller. R2 controls the slope of the bus driver transitions. CAN channel two is identical to channel one, except for three read/wife lines. Here, R9 controls the slope of the bus driver transitions*

is sent to a MAX7219 to be displayed on four 7-segment LEDs. These routines run in conjunction with a routine on the PC that monitors the bus for message 5 12 and converts the data from the ADC to millivolts. It then retransmits the millivolt value in message 732. A potentiometer is connected to input zero of the 80C592's ADC. The 80C31 displays the converted voltage.

HARDWARE

The hardware consists of three network nodes (four if you count the second node on the PC). Two of the nodes are implemented using the Philips PCA80C200 CAN controller chips (see Figures 3 and 4), the third with a Philips 80C592 microcontroller (see Figure 5). Available to me were DIP and surface-mount versions of the 80C200. I wanted to test both packages, but the SOIC is not particularly amenable to wire-wrap prototyping. However, with a little wire wrap and some hot glue, an SOIC package becomes a DIP.

Using the 80C200 makes implementing a CAN node so easy it's almost a no brainer. From the processor's point of view, the 80C200 is just 32 bytes of memory and that is how it is interfaced-it's just like a memory device. The 80C200 has a mode-select pin for Intel and Motorola and an ALE pin for processors with multiplexed address and data buses.

Interfacing to an 8031 is too easy. There are no gates or glue logic. You just wire it up like RAM. The 80C31 is implemented in the standard configuration we've all seen and used dozens of times (see Figure 4a). Figure 4b pictures the MAX7219 and four 7-segment LEDs used for user output.

There's also software to convert a signed integer to the correct bitstream and send it to the '7219. The 8-KB ROM and 40-KB RAM memory configuration of the 80C3 1 and 80C592 controller boards (see Figure 5b) stems from my development environment. I use a program loader I developed a few years ago (*INK* 40). The 40 KB of RAM provides prototype RAM for downloading test code. Refer to Figures 4b and 5b for the ROM and RAM memory configurations for the 80C592 and 80C3 1 boards. The memory configurations are identical for both boards.

For the PC, there are a couple of choices. The 80C200 could be interfaced as a simple 32-byte RAM with no fuss and no muss. But given the colored development of the PC architecture, where can you find a 32-byte slot in memory that is guaranteed not to have something residing there?

The next option is to locate in the I/O space. The interface is again straightforward-just map the chip as 32 consecutive I/O addresses and you're done. However, the PC has very limited free I/O space. It is impossible to find 32 free bytes.

I resolve this by mapping access to the chip as an address register and a data register (see Figure 3a). Mapping the chip this way requires a single extra inverter. Reading and writing now becomes a two-step process.

To access the chip, the address of the specific register is written first to the address register. The data is then read from or written to the data register. This technique takes advantage of the built-in ALE capability of the 80C200. As Figure 3b shows, the write strobe line for the address register is connected to the ALE of the 80C200.

The data at the address register is latched into the 80C200 on chip-address register and the next read from or write to the data register accesses the selected memory location/register in the 80C200. For repeated reads from the same register location (e.g., the

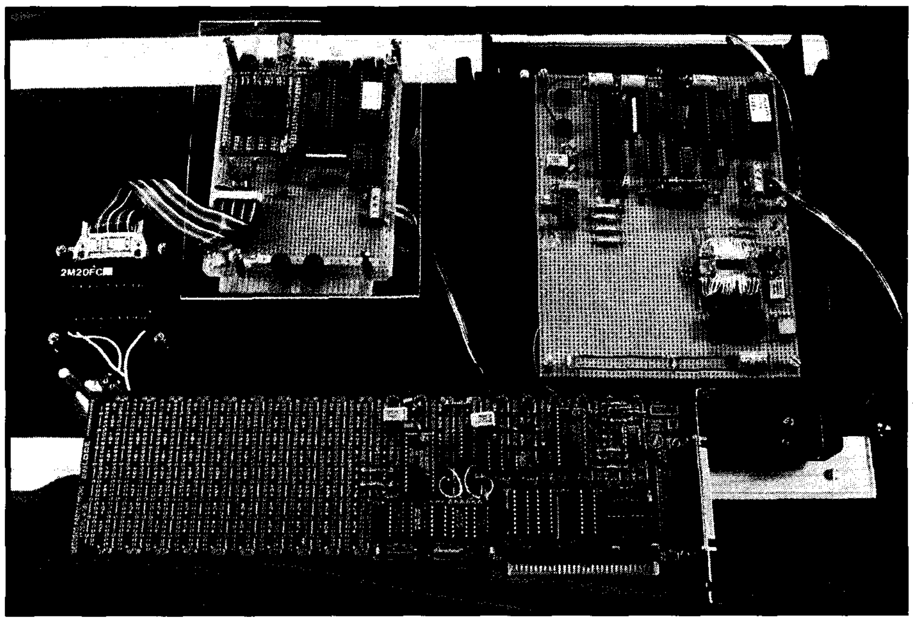


Photo 1—*The network consists of a PC/ISA dual* communications *controller, an 80C31,* and an 8X.592 enhanced controller.

status register), the address does not need to be written each time.

With the 80C592, there is no hardware interface to worry about. The CAN controller is built into the chip. There are multiple methods to access the register set of the onboard CAN controller including autoincrementing register pointer, direct memory access, and address/data register set. It turns out that using the address/data register set to access the controller maps identically to the method used for the PC. It just gets easier and easier.

CAN controllers require a local clock to set bit timing. CAN specifies a maximum bit rate of 1 MHz. Each CAN controller provides a binary divide-by register to divide the local clock to the network bus rate.

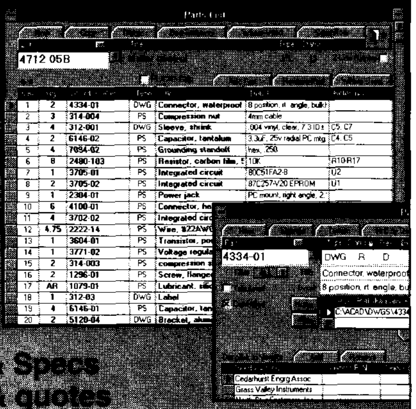
I wanted to clock the network at 1 MHz and I had a handful of 12-MHz crystals, so I put two of the crystals on the PC board and another on the 80C3 1. When I got around to the 80C592, I hit a problem. The onboard CAN controller pulls its clock from the microcontroller clock. However, the onboard UART also uses the microcontroller clock. I wanted to communicate with the UART at 9600 bps. The CAN bus bit rate needed a crystal with a multiple of 1 MHz while the UART bit rate needed a crystal with the well-known 11.0592 MHz. One of the two bit rates was going to be off.

I ended up using the 11.0592 MHz and dividing by eleven for the CAN bit

This Parts List Software is for Engineers / Designers

Not MRP: P&V actually assists development!

Windows™ program helps you stay organized



4712 05B

1	2	4334-01	DWG	Connector, waterproof	8 position, rt angle, bulkh	
2	3	314-004	PS	Compression nut	4mm cable	
3	4	312-001	DWG	Sleeve, shrink	.004 vinyl, clear, 7 3 ID ±	C5, C7
4	2	6146-02	PS	Capacitor, tantalum	3.3uF, 25v radial PC mtg	C4, C5
5	4	7094-02	PS	Grounding standoff	hex, .250	
6	8	2480-103	PS	Resistor, carbon film, 5	10K	R10-R17
7	1	3705-01	PS	Integrated circuit	80C51FA2-B	U2
8	2	3705-02	PS	Integrated circuit	87C257-V20 EPROM	U1
9	1	2304-01	PS	Power jack	PC mount, right angle, 2.	
10	6	4100-01	PS	Connector, he		
11	4	3702-02	PS	Integrated circ		
12	4.75	2222-14	PS	Wire, #22AWG		
13	1	3604-01	PS	Transistor, po		
14	1	3771-02	PS	Voltage regula		
15	2	314-003	PS	compression s		
16	2	1296-01	PS	Screw, flange		
17	AR	1079-01	PS	Lubricant, sili		
18	1	312-03	DWG	Label		
19	4	6146-01	PS	Capacitor, tan		
20	2	5120-04	DWG	Bracket, alum		

4334-01

DWG R D

Connector, waterproof

8 position, rt angle, bu

C:\ACAD\DWGS\4334

Cedarhurst Engrg Assoc

Grass Valley Instruments

parts lists for products in development.

Keep track of:
-Drawings & Specs
-Suppliers & quotes
-Product material costs
-Engineering stock

Parts&Vendor$™

...for independent and departmental projects. Use alone or in a workgroup.

$99 + shpg **Call 800.280.5176**

Trilogy DESIGN™ voice.916.273.1985 fax.916.477.9106 P.O. Box 2270, Grass Valley, CA 95945

Requires 486, 8meg min. RAM, Windows™

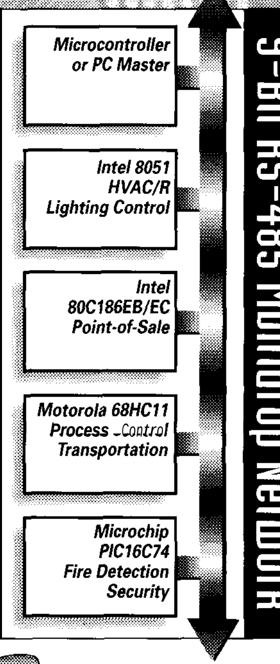
#117

Microcontroller Networks (µLANs)

With Cimetrics' 9-Bit µLAN you can link together up to 250 of the most popular 8- and 16- bit microcontrollers (8051, 80C196, 80C186EB/EC, 68HC11, 68HC16, PIC16C74).

The 9-Bit µLAN is:

- ***Fast***—A high speed (62.5k baud) multidrop master/stave RS-485 network
- ***Flexible***— compatible with your microcontrollers
- Reliable-robust IS-bit CRC and sequence number error checking
- Efficient-low microcontroller resource requirements (uses your chip's built-in serial port)
- *Friend/y-* Simple to use C and assembly language software libraries, with demonstration programs
- ***Complete***— includes network software, network monitor and RS-485 hardware
- Practical-applications include data acquisition and distributed control


Microcontroller or PC Master
Intel 8051 HVAC/R Lighting Control
Intel 80C186EB/EC Point-of-Sale
Motorola 68HC11 Process Control Transportation
Microchip PIC16C74 Fire Detection Security
9-Bit RS-485 Multidrop Network

Cimetrics TECHNOLOGY

Cimetrics Technology • 55 Temple Place • Boston, MA 02111-1300 • Ph617.350.7550.Fx617.350.7552

#118

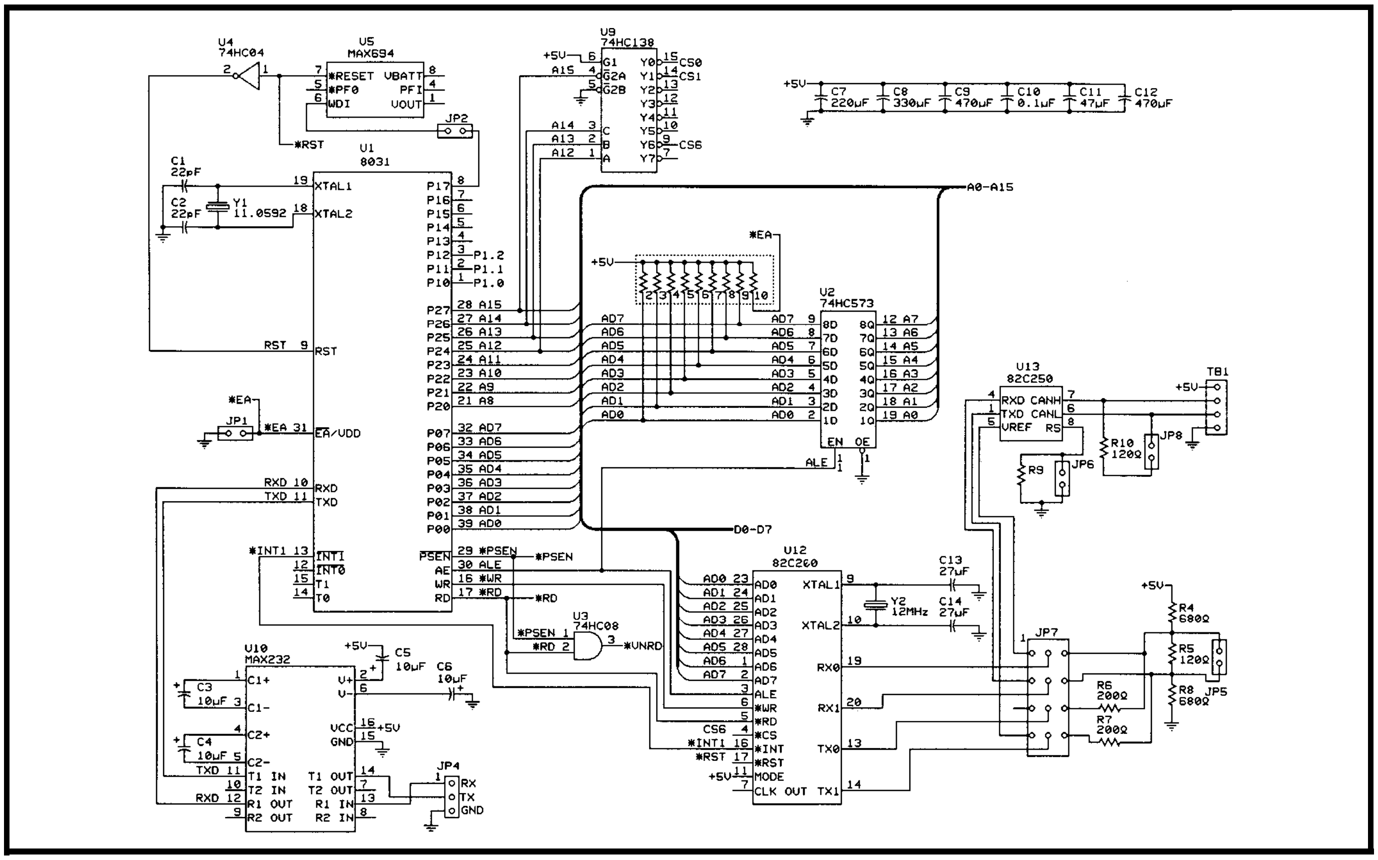
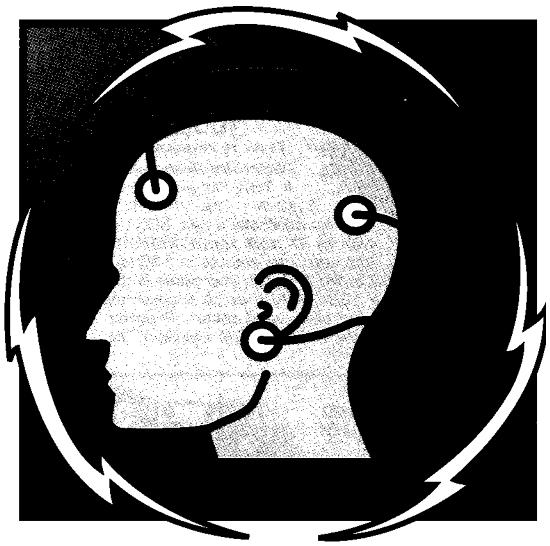


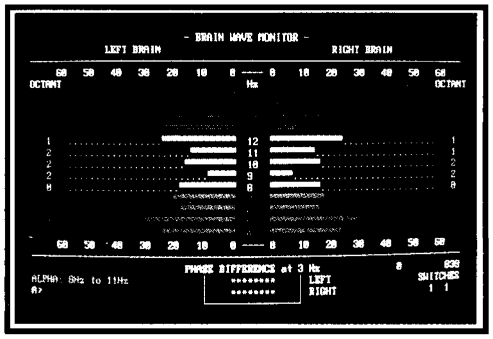
Figure 4a—*The* **controller** *board relies on a standard 80C31 for message handling and process control. The interface to the CAN bus uses a Philips PCA80C200. R9 controls the slope of the bus driver transitions.*

HAL-4

EEG Biofeedback Brainwave Analyzer



The HAL-4 kit is a complete; battery-operated 4-channel electroencephalograph (EEG) which measures a mere 6" x 7". HAL is sensitive enough to even distinguish different conscious states-between concentrated mental activity and pleasant daydreaming. HAL gathers all relevent alpha, beta, and theta brainwave signals within the range of 4-20 Hz and presents it in a serial digitized format that can be easily recorded or analyzed. HAL's operation is straightforward. It samples four channels of analog brainwave data 64 times per second and transmits this digitized data serially to a PC at 4800 bps. There, using a Fast Fourier Transform to determine frequncy, amplitude, and phase components, the results are graphically displayed in real time for each side of the brain.

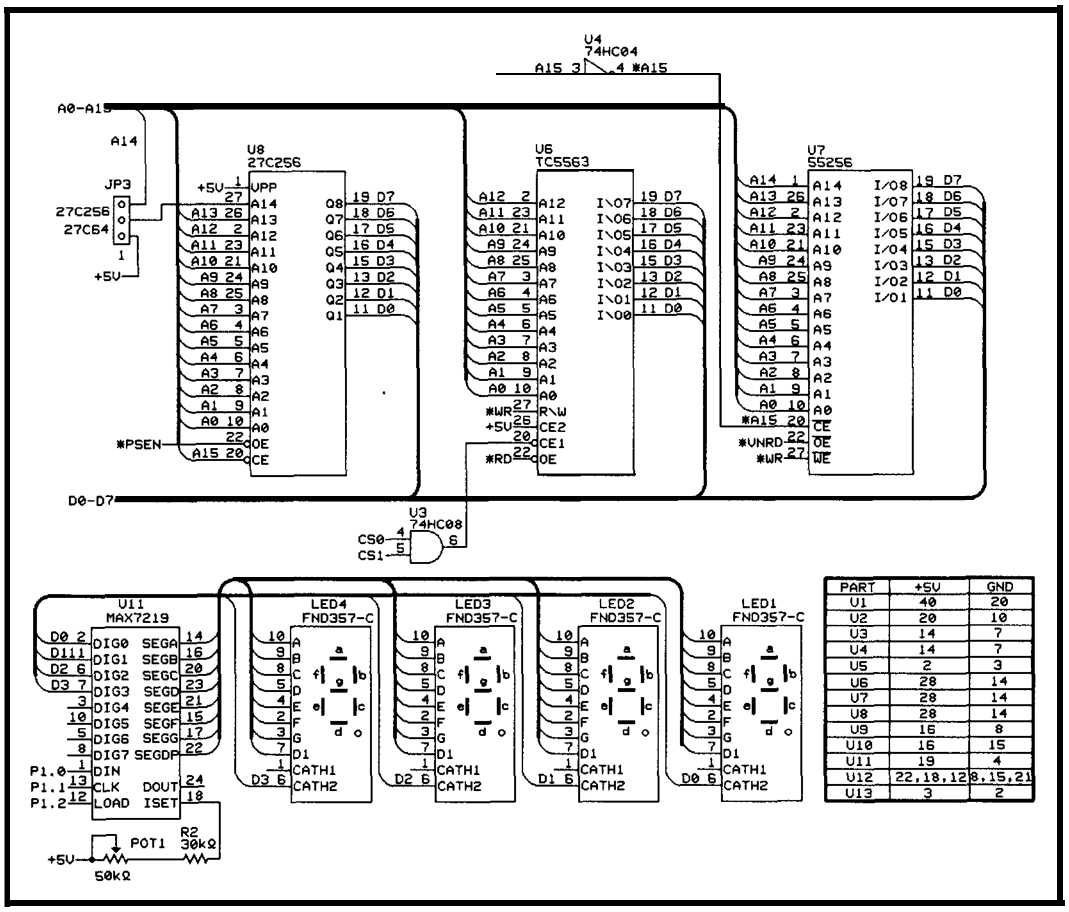

```
                    - BRAIN WAVE MONITOR -
        LEFT BRAIN                          RIGHT BRAIN
  60  50  40  30  20  10  0 ---- 0  10  20  30  40  50  60
OCTANT                       Hz                        OCTANT
1                            12                            1
2                            11                            1
2                            10                            2
2                            9                             2
8                            8                             8
  60  50  40  30  20  10  0 ---- 0  10  20  30  40  50  60
               PHASE DIFFERENCE at 3 Hz        8      030
ALPHA: 8Hz to 11Hz         ********  LEFT         SWITCHES
A>                         ********  RIGHT          1 1
```

HAL-4 KIT...... NEW PACKAGE PRICE - $279 +SHIPPING
Contains HAL-4 PCB and all circuit components, source code on PC diskette, serial connection cable, and four extra sets of disposable electrodes.

to order the HAL-4 Kit or to receive a catalog,
CALL: (203) 875-2751 OR FAX: (203) 872-2204

CIRCUIT CELLAR KITS • 4 PARK STREET
SUITE 12 • VERNON • CT 06066

• The Circuit Cellar Hemispheric Activation Level detector is presented as an engineering example of the design techniques used in acquiring brainwave signals. This Hemispheric Activation Level detector is not a medically approved device, no medical claims are made for this device, and it should not be used for medical diagnostic purposes. Furthermore, safe use requires HAL be battery operated only!



PART	+5V	GND
U1	40	20
U2	20	10
U3	14	7
U4	14	7
U5	2	3
U6	28	14
U7	28	14
U8	28	14
U9	16	8
U10	16	15
U11	19	4
U12	22,18,12	8,15,21
U13	3	2

Figure 4b—*The* controller board relies on a *MAX7219 LED* display driver *for visual* display of *information.*

rate. This gives about a 0.5% timing error in the CAN bitstream. I am using the system at relatively low packet rates (about ten packets per second) and l-MHz bit rate. I have yet to have a packet error. Your mileage may vary.

BUS PHYSICAL INTERFACE

CAN supports a variety of physical interfaces to the bus:

- a simple two resistors per node plus termination
- transformer coupled
- optoisolated
- differential pair
- and on and on

I implemented the simple resistor interface and the differential pair using the Philips PCA82C250 (see Figures 3b, 4a, and 5b).

The simple resistor interface provides a low-cost compact interface, but has a more limited driving capability. The 82C250 features differential pair driving (similar to RS-485), edge-slope control, a $V_{REF}/2$ output, bus

TURBO-128

THE *NEXT GENERATION* EMBEDDED CONTROLLER

*** * NO DEVELOPMENT TOOLS REQUIRED * ***

READY TO PROGRAM IN BASIC OR ASSEMBLY

Photronics Research introduces the T-128: A True Single Board BASIC Development System The T-128 is based on Dallas Semiconductor's new 8051-compatible DS80C320 With its 2X clock speed (25MHz) and 3X cycle efficiency, an instruction can execute in 160ns: an 8051 equivalent speed **of 62.5MHz!!!** Equally impressive is the T-l 28's high-speed NVRAM interface. Any of the 128K RAM may **be programmed directly from a** PC file **through the console: eliminating** EPROMs and associated tools. Program Development has never been faster or more convenient. even with the finest EPROM emulator. The T-128 features PORT 0 bias and EA-select for DS87C520 upgrade.

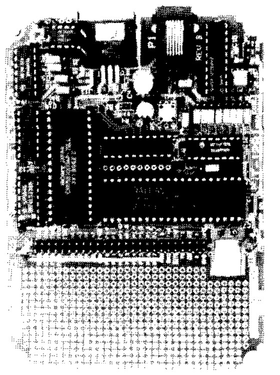
PROCESSOR

*Dallas Semiconductor's DS80C320
- 3uxz more efficient than the 8051
- Three 16-bit Timer/Counters
- 1 3 Interrupts (6 Ext. 7 Int)
- A second 1 6-bit Data Pointer
- 3a4 Bytes of Internal RAM

*Programmable Watchdog
- Brownout Protection
- Powerf ail Reset/Interrupt
- Power-On Reset
- Fully supported by ***Franklin C51***

MEMORY

- Entire 12aK Memory Map populated with *fast* NVRAM (64K DATA + 64K CODE)
- All memory programmed on-board
- Partitionable as CODE/DATA/OVERLAID
- Code Space is Write-Protectable
- State-of-the-Art Data Protection



Only 5" x 3-3/4" • 500-hole Proto Area • Console/Power connected by a single 4-conductor telephone wire (very convenient)

BASIC520
- Modified BASIC-52 Interpreter (BASIC-520) Now Fast Enough for New Applications

*Stack BASIC Programs and Autorun

*CALL ASM Routines for Maximum Speed

I/O
- Three abii Parallel Ports
- Two Full-Duplex RS232 Serial Ports

*Decoded Device I/O Strobes
- 5&% Bus Connector

UPGRADE
- DS87C520 processor (33MHz)
- Instruction cycle. 121ns
- 8.25 MIPS
- 8051 equivalent: 82.5 MHz
- Internal 16K ROM/l K SRAM

Comas Ready to Run with power adapter/cable assembly. Includes utility diskette with DETAILED TECHNICAL **MANUAL**

$199 in **QTY.**

109 Camille St. • Amite, LA 70422 • (504) 748-9911 • Tech Support (504) 748-7090 • FAX (504) 748-4242

#120

tristate (when powered down), and an ISO 11898-compatible interface.

Through this method, over one hundred 82C250s can be connected to a single bus. The problem with the 82C250 is that it is about as easy to get as a hen's teeth. I waited eight weeks to get a sample part when the typical wait from Philips is about a week. Maybe supplies will have loosened by the time this gets to press.

82C250 SLOPE CONTROL

One feature of the 82C250 is the ability to limit the rise rate of the bus transitions. Limiting the rise rate helps control transients and generated RFI. The Rs pin (pin 8) offers three modes of operation: high speed, slope control, and standby.

During high-speed operation, the transmitter output transistors are simply switched on and off as fast as possible. In this mode, no measures are taken to limit the rise and fall slope. Shielded cable should be used to avoid RFI problems. The high-speed mode is selected by connecting pin 8 to ground.

When operating at lower speeds or with shorter bus lengths, an unshielded twisted pair or a parallel pair can be used for the bus. To reduce RFI, the rise and fall slope should be limited.

The rise and fall slope can be programmed with a resistor connected from pin 8 to ground. The slope is proportional to the current at pin 8. If a high level is applied to pin 8, the transceiver enters a low-current standby mode. In this mode, the transmitter is switched off and the receiver is switched to a low current.

When dominant bits are detected on the bus, RxD switches to a low level. The microcontroller should react to this condition by switching the transceiver back to normal operation via pin 8. Because the receiver is slow in standby mode, the first message is lost.

CONCLUSION

If you are interested in learning about CAN, the texts in the reference section of this article and its prequel (*INK* 58) would be a good place to start. Also, contacting the field engineers at Philips, Intel, or Motorola should produce a ream of references.

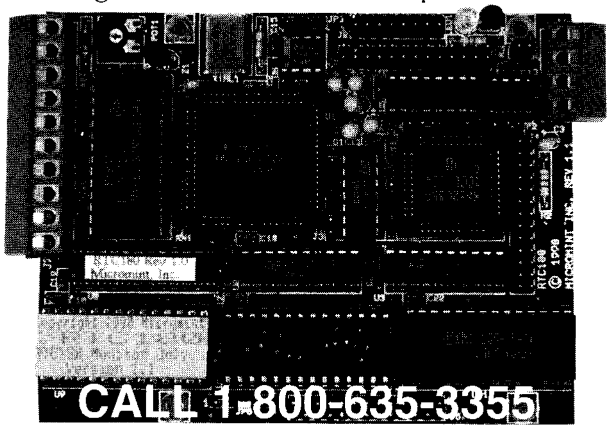
If you can't wait to start experimenting with CAN, I recommend dual PC-hosted controllers. PC software tools are flexible and powerful. Working in the PC environment provides tools to easily peek, poke, and prod your design before migrating to an embedded system where the tools might not be so capable.

Placing two communications controllers in the PC, whether on the same or separate boards, enables messages to be passed and the communications performance evaluated quickly and at low cost. DIP, a third-party company that advertises in ***Circuit Cellar INK,*** is selling a single-channel PC-CAN controller for under $200.

If you want a microcontroller-based solution, there are numerous third-party solutions and evaluation boards available from the silicon vendors. Philips has an inexpensive 80C31 plus 80C200 evaluation board.

STOP LOOK LISTEN!

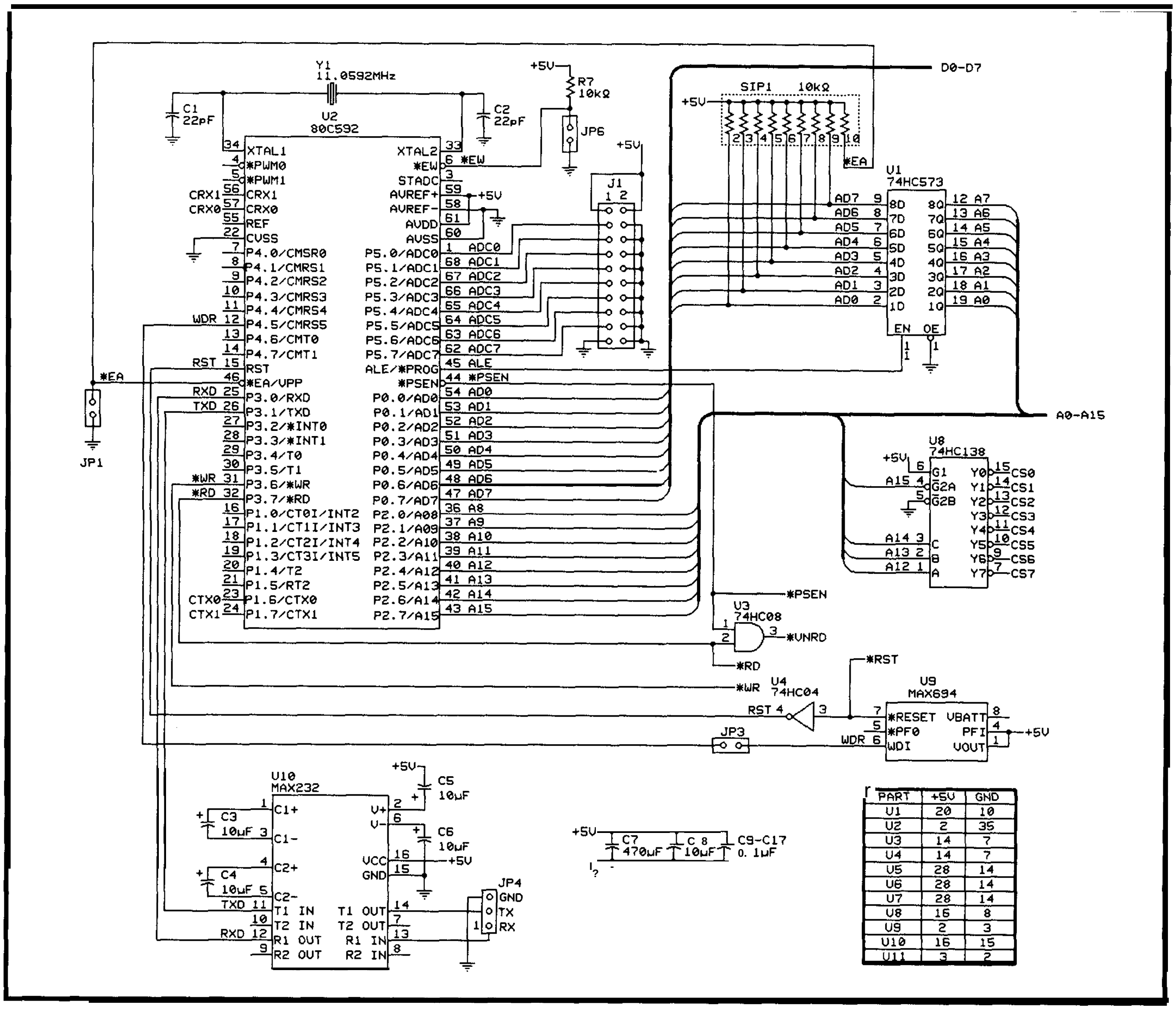
Odds are that some time during the day you will stop for a traffic signal, look at a message display or listen to a recorded announcement controlled by a Micromint RTC180. We've shipped thousands of RTC180s to OEMs. Check out why they chose the RTC180 by calling us for a data sheet and price list now.



CALL 1-800-635-3355

MICROMINT, INC.
4 Park Street, Vernon, CT 06066
(203) 871-6170•Fax (203) 872-2204

in Europe: (44) 0285-658122•in Canada: (514) 336-9426•in Australia: (3) 467-7194•Distributor Inquiries Welcome



PART	+5V	GND
U1	20	10
U2	2	35
U3	14	7
U4	14	7
U5	28	14
U6	28	14
U7	28	14
U8	16	8
U9	2	3
U10	16	15
U11	3	2

Figure **5a**—The Philips 80C592 is an enhanced 80C31 core with an onboard CAN controller, AID converter, PWM, enhanced fimers, and more.

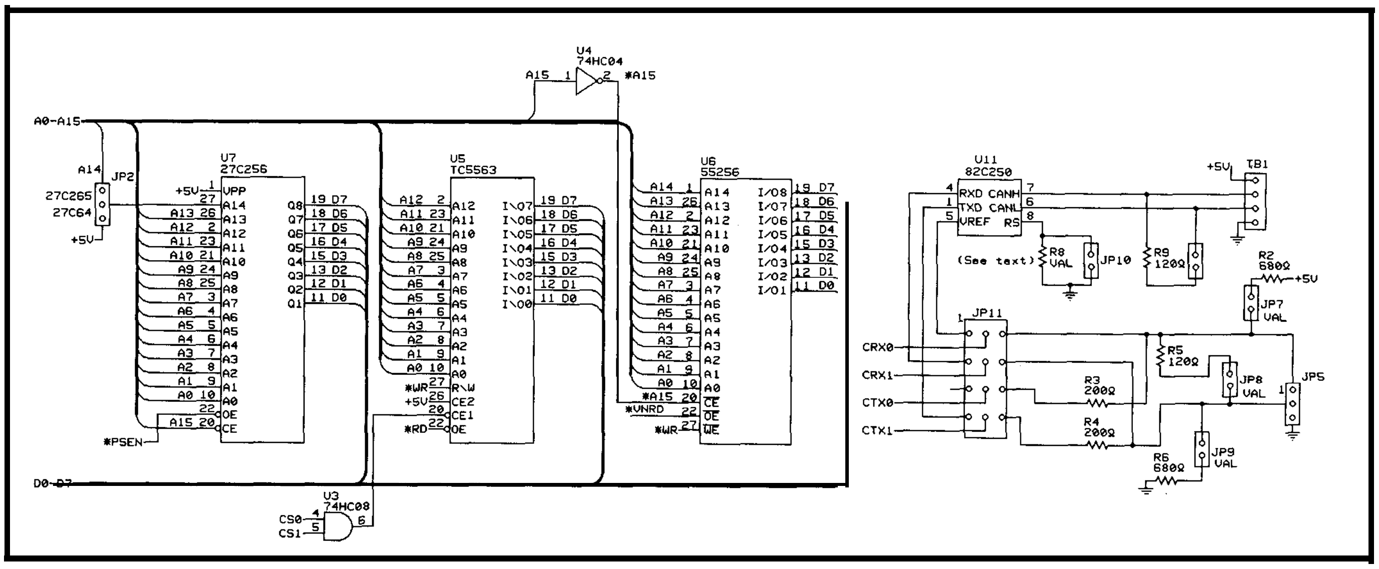


Figure **5b**—The memory configurations for the 80C592 and 80C31 controller boards are identical. Each board sports 8-KB EPROM and 40-KB RAM for data collection and profofyping. The CAN physical interface includes differential driving and simple bus pull-up and pull-down resistors. R8 controls the slope of the bus driver transitions.

If you do decide to build your own controllers and you're planning to use the 82C250, *you* might design in the simple resistor interface so the board can go to testing while you wait for the 82C250s.

Brad Hunting has industrial experience in embedded systems development for equipment automation and process control. He has recently returned to school to complete a graduate degree at Rensselaer Polytechnic Institute. He may be reached at huntib@cat.rpi.edu.

SOFTWARE

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

REFERENCES

J. Brauninger, T. Kuttner, A. Loffler, *Controller Area Network for Truck and Bus Applications* (Warrendale, PC: SAE Transactions 902211, 1990) 1-16.

Intel, 82526 *Serial Communications Controller Architectural Overview, 270678-003 (1992).*

Intel, 82527 *Serial Communications Controller Architectural Overview, 272410-001 (1993).*

Intel, 82527 *Serial Communications Con troller: Con troller Area Network Protocol, 272250-003 (1993).*

Signetics, *80C51-based 8-Bit Microcontroller,* (1992) 480-5 17.

SOURCE

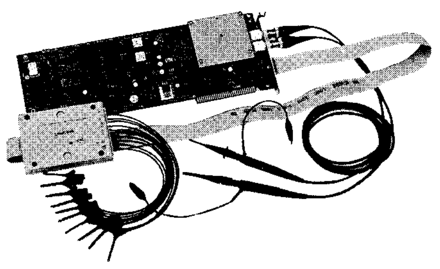
DIP, Inc.
P.O. Box 9550
Moreno, CA 92552-9550
(909) 924-1730
Fax: (909) 924-3359

I R S

410 Very Useful
411 Moderately Useful
412 Not Useful

PC-Based Instruments
200 MSa/s DIGITAL OSCILLOSCOPE

HUGE BUFFER
FAST SAMPLING
SCOPE AND LOGIC ANALYZER
C LIBRARY W/SOURCE AVAILABLE
POWERFUL FRONT PANEL SOFTWARE



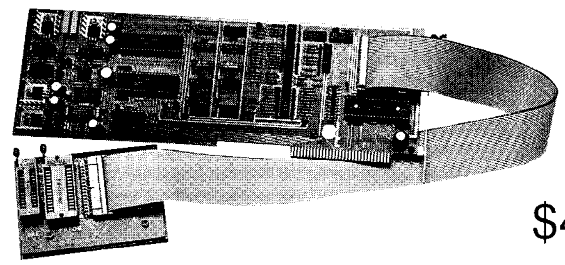
DSO Channels
2 Ch. up to 100 MSa/s
or
1 Ch. at 200 MSa/s
4K or 64K Samples/Ch
Cross Trigger with LA
125 MHz Bandwidth

Logic Analyzer Channels
8 Ch. up to 100 MHz
4K or 64K Samples/Ch
Cross Trigger with DSO

$1799 - DSO-28204 (4K)
$2285 - DSO-28264 (64K)

Universal Device Programmer

PAL
GAL
EPROM
EEPROM
FLASH
MICRO
PIC
etc..

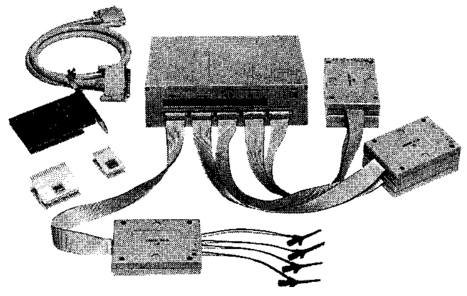


$475

Free software updates on BBS
Powerful menu driven software

400 MHz Logic Analyzer

- up to 128 Channels
- up to 400 MHz
- up to 16K Samples/Channel
- Variable Threshold Levels
- 8 External Clocks
- 16 Level Triggering
- Pattern Generator Option



$799 - LA12100 **(100 MHz, 24 Ch)**
$1299 - LA32200 (200 MHz, 32 Ch)
$1899 - LA32400 (400 MHz, 32 Ch)
$2750 - LA64400 (400 MHz, 64 Ch)

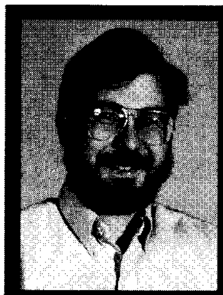
Call (201) 808-8990
Link Instruments
369 Passaic Ave, Suite 100, Fairfield, NJ 07004 fax: 808-8786

#121

How the PC Keyboard Got its Bits

FIRMWARE FURNACE

Ed Nisley



Ed takes a brief aside from protected-mode programming to look at the keyboard. After reviewing the history of keyboard development, Ed gets into the nitty-gritty issues of keyboard microprocessors and scan codes.

The only thing new in the world is the history you don't know.
Harry Truman

History is more or less bunk.
Henry Ford

Truman is a better guide than Ford when you deal with the innards of PC hardware. As we've seen many times in this column, even the most bizarre PC designs made perfect sense at the time. Yes, with years of hindsight, we can all do a better job. But, I doubt I would have done as well back then.

The IBM PC's keyboard facilities are a case in point. The closer you look, the more bizarre they become. Three separate processors, a bidirectional serial interface, read- and write-only ports, mysterious status and control bits, overlapping key codes, and a wealth of rarely used options all cry, "Kludge!" at the top of the data sheets.

After I collected all my notes for this topic, I realized that the subject is incomprehensible without more background than fits in a single column. This month, I'll explain how the keyboard got into its present condition and make Demo Taskette 3 in the FFTS kernel display the default system scan codes. Next month, we'll delve into unexplored territory.

So, let us begin...

IN THE BEGINNING

Contrary to what many people think, the Original IBM PC wasn't the first product the IBM Corporation produced. The PC designers ran a classic Skunk Works operation, drawing whatever they could use from the rest of the company and ignoring whatever didn't fit. As a result, the PC contained some things old, some things new, some things borrowed, and yes, some things Blue.

The PC keyboard's pedigree, in particular, stretches back to the legendary IBM Selectric and the infamous O-series keypunches. Among other advances, IBM honed capacitive-sensing key technology to a fine edge; the bold tactile click you either loved or hated was designed in, not added on. The only catch was the manufacturing cost-if you ever see an old PC keyboard, pry off a keycap and admire the small parts!

In any event, the keyboard had three main sections: typewriter keys in the middle, 10 function keys to the left, and a block of keys to the right that served for both cursor control and numeric data entry. Each key had an identification number, assigned more or less left to right and top to bottom within its group.

An Intel 8048 microcontroller scanned the capacitive key matrix and sent a scan code to the PC when each key was pressed. Believe it or not, each key's scan code exactly matched its key number. The Esc key in the upper-left corner of the main typewriter area was Key 1 and returned scan code 01. Key numbers are in decimal and scan codes are in hex just to keep you on your toes.

Each time a key went down [a *key make* in IBM parlance), the 8048 stored its scan code in a 16-entry buffer. All keys had typematic action. In other words, if the key remained down, the 8048 stored an additional make code after about 500 ms and another code every 100 ms after that. When the key went up (a *key break),* the 8048 set the high bit of the key's scan code and stored it in the buffer. The Esc key's break code was 81 hex.

Figure 1 shows the overall scheme. Makes a lot of sense so far, doesn't it?

The keyboard transmitted scan codes from the buffer to the PC

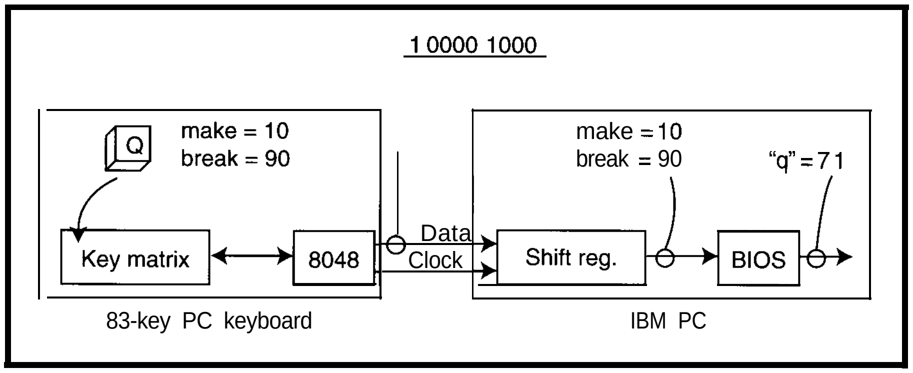


Figure I--The Original IBM *PC 83-key* keyboard *scan* codes *were simply the* key numbers, *assigned left to right and fop to bottom. The break code for each key was equal to the scan code with bit 7 sef high. A high start bit preceded each code through the serial interface and triggered an interrupt when* if emerged from *the* shift *register inside the PC. The BIOS translated the* scan codes *info a standard set of key codes that depended on the shift keys.*

through a four-wire cable: +5 V, ground, Data, and Clock. A fifth line, -Keyboard Reset, wasn't used and vanished from later specs. Clock and Data were open-collector TTL lines, nominally bidirectional, but in point of fact, the PC didn't have much to say to the keyboard.

The PC's keyboard interface was charmingly simple: an 8-bit shift register driven by the keyboard's Clock and Data lines. The 8048 sent a "1" start bit before clocking eight scan code bits into the shift register. When the start bit emerged from the shift register, it set a latch that triggered IRQ 1 and pulled the keyboard Clock line low to prevent further transmissions. After reading the shift register, the PC toggled an I/O port pin that cleared the shift register, reset the interrupt, and enabled the keyboard's Clock line in preparation for the next scan code.

Unlike many keyboards of the time, the PC keyboard had n-key rollover. You could, for instance, hold down the Ctrl, Alt, and Del keys at once and be sure the BIOS would receive each scan code in the correct order with typematic action intact. Most key chords didn't make much sense, but a TSR cottage industry arose around the ability to detect finger exercises such as left Shift, right Shift, Ctrl-S.

The PC's BIOS keyboard functions, unlike those for the serial port and printer, were complete enough that nearly all programs used them. Indeed, to this day many programmers are blissfully unaware of how the BIOS goes about its task. We embedded folks don't have that luxury. In the Protected Land, you can see all the way down to bare silicon and copper!

The BIOS tracked the make and break codes for each shift key to figure out what to do with all the other keys. For example, the BIOS translated Key 16 (make code 10 and break code 90) into character "q" (71 hex], "Q" (51), or Ctrl-Q (11) as appropriate. The Read Keyboard Input BIOS function returned the translated character in the AL register along with the key's scan code in AH to uniquely distinguish the key. The notation 10/71,10/51, and 10/1 1 seen in the references should make more sense now.

The PC keyboard also included an Alt shift key. The BIOS translation for Alt-shifted keys returned zero in AL and the key scan code in AH: 10/00 for Alt-Q. These keystrokes were usually interpreted as special commands rather than characters, but that was up to the program. Alt applied to several other noncharacter keys and the BIOS suppressed the remainder.

Function, cursor control, and special keys behaved much like Alt-shifted keys. The BIOS returned zero in AL with AH containing the raw scan code, a modified scan code unique to the key combination, or a duplicate of another key's character. The 7-Home key (Key 71, scan code 47), returned 47/00,47/37, and 77/00 for normal, Caps, and Ctrl shifts. Alt-Home was suppressed. The key codings show that Caps-Home is ASCII character "7" and Ctrl-Home is entirely unique.

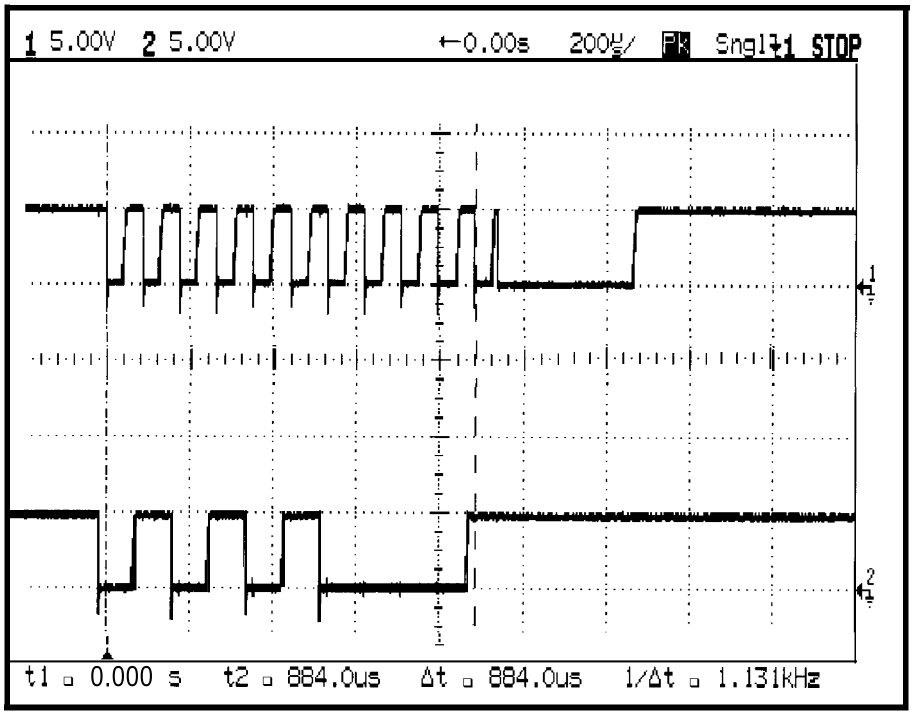


Photo 1—*This scope shot shows the bits flowing through the keyboard cable* after *pressing the Q key on a 101-key Enhanced keyboard. The falling edges of the clock waveform in Trace 1 mark valid data bits in Trace 2. The start bit, identified by fhe left cursor, is always a low level, followed by fhe eight data bits for make code 15 hex, an* ***odd*** parity *bit, and a high stop bit, marked by the right cursor. The 8042 system keyboard controller inside the PC pulls the Clock* line *low after it receives the stop bit to* prevent *any further transmissions until the key is processed.*

Although there are some quirks, on the whole, the scheme makes a lot of sense. Using simple hardware, mapping various shift states into ASCII characters, providing an Alt shift case to expand the codes, and making everything reasonably easy was, ahem, the PC's key to success.

The PC Compatibility Barnacles began growing seconds after the Original PC hit the store shelves.

CONVERTED CODES

The IBM PC/AT (unofficially: "Advanced Technology") ushered in the '286 CPU, protected mode with 64-KB segments, and the all-new, ergonomically advanced, 84-key keyboard.

The 84th key sported a Sys Req label and was supposed to trigger special operating-system functions. PC-DOS (the only operating system that mattered) had no such functions. Application programs ignored it, hard-core utilities put it to a variety of peculiar tasks, and most folks never noticed it.

The Original PC's 83-key keyboard featured sculptured key caps. Although they looked nice, their main function was ensuring that you pressed each key directly over its switch mechanism because the keys jammed if you hit them anywhere else. PC/AT keys used a different mechanical design that stabilized the keys and allowed big, flat key caps. It also had a *slightly* different layout.

The entire numeric keypad slid half an inch to the right and captured the Esc key in its upper-left corner. The open single quotation mark and tilde ('~) key moved from beside the left-Shift key to snuggle in the crook of an L-shaped, aircraft-carrier-sized Enter key. Caps Lock and Ctrl swapped places, much to the relief of Selectric owners and the disgust of everybody else. Another cottage industry sprang up delivering TSRs to unswap the offending keys.

Under the covers, though, every*thing* changed as IBM renumbered the keys to match their new locations. This made some sense, even though there were some peculiarities. The PC function keys bore labels F1 through F10 in left-to-right, top-to-bottom order, but had key numbers 65-74 in top-to-bottom, right-to-left sequence. The Esc key, formerly Key 1, became Key 90 in its new location, leaving a suspicious 15-key gap after F9. PC pundits wondered what IBM was up to-surely, those missing key numbers meant something.

As the PC and later the AT became a roaring success, IBM began "converging" their myriad disparate keyboard lines into a single offering. The PC keyboard's method of generating break codes by setting bit 7 of the make code limited the number of distinct keys to 128, and IBM realized that a mere 128 keys might not suffice for some mainframe or minicomputer keyboards. Thus, the break codes on the PC/AT's keyboard became two-byte sequences: FO followed by the key make code.

For example, the Q key, formerly Key 16, became Key 17 with make code 15 and break code FO 15. Just to hammer the point home, Sys Req was Key 105 with make code 84 (note the high-order bit!) and break code FO 84. Perforce, the new keyboard was entirely incompatible with the old PC.

Changing the PC's barnacle-encrusted scan codes would shatter every existing PC program. The PC/AT got around that problem by translating the new keyboard scan codes into the familiar system scan codes everyone knew and loved. As far as application programmers were concerned, the new AT keyboard was identical with the old PC keyboard,

Acronyms

CPL	Current Privilege Level
DPL	Descriptor Privilege Level
EOI	End Of Interrupt (command)
FDB	Firmware Development Board
FFTS	Firmware Furnace Task Switcher
GDT	Global Descriptor Table
GDTR	GDT Register
IDT	Interrupt Descriptor Table
IF	Interrupt Flag
IMR	Interrupt Mask Register
IOPL	I/O Privilege Level
LDT	Local Descriptor Table
LDTR	LDT Register
NT	Nested Task
P bit	Present bit (in a PM descriptor)
RF	Resume Flag
RPL	Requestor Privilege Level
TF	Trap Flag
TR	Task Register
TSS	Task State Segment

although the new distinction between keyboard and system scan codes remains obscure to this day.

Scan code translation, however, could not occur at the BIOS level because some ill-behaved programs burrowed through the BIOS directly to the hardware interface. IBM solved that problem by interposing yet another microcontroller between the BIOS and the keyboard cable. Once you have a microcontroller on board, a lot of other things are possible.. .

PASSING THE BUCK

The old PC keyboard sent a start bit followed by eight data bits. The start bit was a 1 that triggered IRQ 1 when it popped out of the system board's shift register. The new system keyboard controller, an Intel 8042, had enough intelligence to support an entirely different keyboard serial protocol.

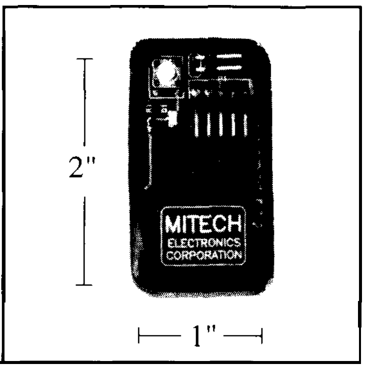
The PC/AT keyboard used a data format much like ordinary RS-232: one start bit (a zero), eight data bits, an odd parity bit, and one stop bit. Unlike RS-232 asynchronous data, a clock signal indicated when each data bit was valid. Photo 1 shows this interface in action. The enhanced keyboard that I'll discuss shortly uses the same signaling technique. The two micro-controllers verify parity and request retransmissions as needed, eliminating most system error handling.

In addition to an 84th key, the PC/AT keyboard also sprouted three indicator LEDs: Caps Lock, Numeric Lock, and Scroll Lock. Nobody in the PC world knew what to do with the latter, but the former two eliminated a cottage industry of on-screen "video LED" indicators. For the first time, the system had something to say to the keyboard, as the three LEDs are controlled by the BIOS rather than directly by the keyboard.

The AT's designers also gave the new system keyboard controller life-and-death responsibility: one of the 8042's output pins drove the system's Reset circuit. As I described in INK 38, resetting the entire system was the only way to get the 80286 CPU out of protected mode. Thus, was born the Worst Hack in PC-dom.

OUR SMALLEST EPROM EMULATOR

Eliminate the need to burn and ***learn***
all in a package about the size of a 9 volt battery



- Super small (about 2"x1"xl ")
- Uses Surface Mount Design
- Fits in EPROM socket
- Downloads in less than 7 seconds
- Emulates 2764, 27128, 27256, 275 12
- Access time < 1 OONS
- Plug and Play: Plug cable into computer and download. Once power is supplied to circuit, cable can be removed
- Includes software for IBM compatable PC's
- Loads Intel Hex and Binary files
- Includes Serial Cable & Battery Backup
- Nearly same footprint as EPROM

MITECH
ELECTRONICS
CORPORATION

411 Washington Street,
Otsego, Michigan 49078
TEL: 616-694-4920 FAX: 616-692-2651
Since 1985

CALL NO Wfor Your Special Introductory Price of $194.95

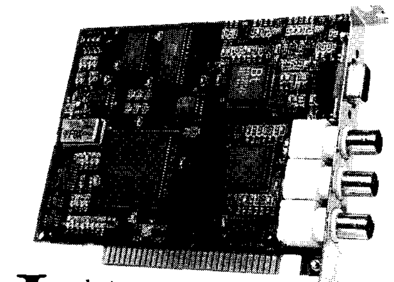
plus $5.00 shipping and handling

VISA & MASTERCARD ACCEPTED

2

PRECISION FRAME GRABBER

FOR ONLY $495*



— CXlOO FEATURES —
. Crystal Controlled Image Accuracy
. Memory Mapped, Dual-Ported Video RAM
. Programmable Offset and Gain
. Input, Output and Overlay LUTs
. Resolution of 5 12x486 or Four Images of 256x243 (CCIR 512x512 & 256x256)
. Monochrome, 8 Bit, Real Tie Frame Grabs
. Graphics Overlay on Live or Still Images**
. External Trigger Input
. RGB or B&W, 30 Hz Interlaced Display
. NTSC/PAL Auto Detect, Auto Switch
. VCR and Resettable Camera Compatible
. Power Down Capability
. BNC or RCA Connectors
. Built-In Software Protection**
. 63 Function C Library with Source Code
. Text & Graphic Library with Source Code
. Windows DLL, Examples and Utilities
. Software also available free on our BBS
. Image File Formats: GIF, TIFF, BMP, PIC, PCX,TGA and WPG

Introducing the CX100 precision video frame grabber for OEM, industrial and scientific applications. With sampling jitter of only ±3 nS and video noise less than one LSB, ImageNation breaks new ground in imaging price/performance. The CXlOO is a rugged, low power, ISA board featuring rock solid, crystal controlled timing and all digital video synchronization.
A Software developers will appreciate the simple software interface, extensive C library and clear documentation. The CXlOO is a software compatible, drop-in replacement for our very popular Cortex I frame grabber. A Call today for complete specifications and volume pricing.

ImageNation Corporation
Vision ***Requires Imagination***
800-366-9131

** THESE OPTIONS AVAILABLE AT EXTRA COST.
* $495 IS DOMESTIC. OEM SINGLE UNIT PRICE

P.O. BOX 276 BEAVERTON, OR 97075 USA PHONE (503) 641.7408 FAX (503) 643-2458 BBS (503) 626.7763

23

In any event, the 8042 translated the new keyboard make and break codes into the new, yet at the same time old, system scan codes. The new Q key returned system make code 10 and break code 90, just like the old Q. The BIOS translation was essentially unchanged because the system scan codes were identical after the 8042 got through with them. Figure 2 shows the successive translations from keycap to BIOS.

About two years after the AT's introduction, IBM dropped another plank: the 101-key Enhanced keyboard. If adding the 84th key was a challenge, imagine seventeen more!

ENHANCED AND ENLARGED

The Enhanced keyboard was known internally as the *Converged* keyboard because it offered all the keys and functions required by IBM's mainframe, minicomputer, and PC divisions. Early users quickly dubbed it the *Concatenated* keyboard in honor of its size and committee ancestry.

The numeric pad drifted another two inches to the right and spat out the Esc key. The empty space refilled with ten dedicated cursor movement, screen control, and editing keys numbered from Key 75 through 89 with a few gaps. Ten function keys fissioned to twelve, migrated en masse to a straight row above the typewriter keys, and became Keys 112 through 123. The Esc key, now Key 110, returned to the far upper-left corner of the keyboard.

PC users wondered why they needed two each of the Ctrl and Alt keys, particularly as the left Ctrl key obstinately remained where Caps Lock should be. Mainframe and minicomputer users, faced with different keycap legends, rejoiced as the Enter key returned to its proper spot below right Shift and worried themselves not at all over broken symmetry.

Moving the function keys gave the keyboard cottage industry a shot in the arm. Giant clone keyboards heaved into view with ten function keys to the west and another dozen up north. Some sported an additional dozen function keys to match the minicomputer layout. Small rodents skittering about the desktop harassed these behemoths, many succumbed to terminal coffee and soda ingestion, and users with no remaining desktop space dealt a death blow.

[The current crop of twisted and bloated ergonomic keyboards are roughly the same size. One wonders if desks have become any larger.. .}

As with the PC/AT's 84-key keyboard, the new Enhanced keyboard was incompatible with all previous PC systems. The Compatibility Barnacles once again dictated that existing programs must work without changes, while new programs should detect and use the new keys.

The new keys between the typewriter keyboard and the numeric pad were not, strictly speaking, new. The numeric pad on 83- and 84-key keyboards served for both cursor movement and numeric entry. For example, pressing the 8-↑ key produced "8" (38 hex) in Num Lock mode and moved the cursor upward in unshifted mode. Heavy-duty spreadsheet users developed finger cramps from shifting and unshifting their way from cell to cell.

The enhanced keyboard separated those functions: the new ↑ key moved the cursor upward, while 8-↑ performed both functions. With the keyboard in Num Lock mode, the numeric pad really was a numeric pad without requiring any additional shift keystrokes.

Now, the question was: how can the BIOS map two keys into the same value while keeping them distinct? The solution is a simple matter of firmware in three different processors....

SHIFTING SANDS

Obviously, the microcontroller in the keyboard (by now an 805 1 or one of its ilk) must know which key is pressed. It could return the same code for two distinct keys, but that would suppress valuable information. IBM's keyboard designers decided to break new ground: the keyboard would keep track of the current shift state and send different scan codes for the same key.

Keypad 8-↑ is Key 96 on both the 84-key and Enhanced keyboards. The controller sends make code 75 and break code FO 75 for that key regardless of which shift keys are pressed. That much remained unchanged.

The new ↑ atop the "inverted T" of cursor keys is Key 83. In normal, unshifted mode, it returns make code EO 75 and break code EO FO 75. Even though there are fewer than 128 keys, many of the new keys return scan codes with an EO prefix byte. In this case, the base scan code is the same as for Key 96, but that is not a general rule.

Now it gets weird.

The ↑ make code returns EO FO 12 EO 75 when it's pressed after the left

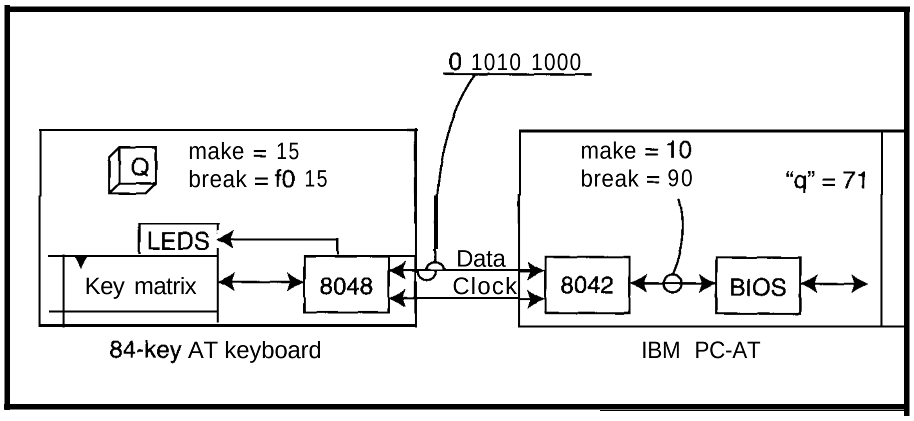


Figure 2—*The IBM* PC/A J *84-key* keyboard used a *completely* different key-numbering *system. The* scan codes *were essentially arbitrary and used all eight data bits. Break codes became a two-byfe sequence: F0 hex followed by the key's scan code. The serial data format used a zero start bit, eight data bits, an odd* parity *bit, and a high stop bit, much like ordinary RS-232 serial communication. An 8042 microcontroller inside the PC/A J handled the new format and translated new scan codes info fhe o/d key codes.*

Listing 1—*This interrupt handler reads system scan codes from fhe 8042 keyboard controller and places them in a ring buffer without analyzing them at all. A matching routine in Demo Task 3 extracts them for display on the VGA. The 8042 converts fhe torrent of keyboard scan codes info the* **slightly** *more familiar system scan codes. The real-mode BIOS processes system scan codes info characters, but those routines are not available in protected mode.*

```
          CODESEG

          PROC    KeyHandler
          USES    EAX,EDX,DS

          MOV     EDX,SYNC_ADDR       ; show a tick
          IN      AL,DX
          OR      AL,40h
          OUT     DX,AL

          MOV     EAX,GDT_DATA        ; get addressability to our data
          MOV     DS,AX

          IN      AL,KEY_DATA         ; read scan code from controller
          Punt
          CMP     [RingCount],RING_SIZE   ; room for one more?
          JE      @@Done              ; nope, bail out

          MOVZX   EAX,AL              ; clear high bytes
          MOV     EDX,[RingHead]      ; aim at head entry
          MOV     [EDX*4 + KeyRingl,EAX   ; save the scan code
          INC     [RingCount]         ; account for it

          INC     EDX                 ; tick and wrap index
          CMP     EDX,RING_SIZE
          JB      @@NoWrap
          XOR     EDX,EDX
@@NoWrap
          MOV     [RingHead],EDX

@Done:
          MOV     AL,NS_EOI           ; now reset the 8259 ISRs
          OUT     I8259A,AL           ; EOI primary controller

          MOV     EDX,SYNC_ADDR       ; remove the tick
          IN      AL,DX
          AND     AL,NOT 40h
          OUT     DX,AL

          POP     DS                  ; restore bystanders
          POP     EDX
          POP     EAX

          IRET                        ; return to interrupted code

          ENDP    KeyHandler
```

Shift (Key 44, scan code 12) is down. The first byte, EO, introduces an extended scan code. The second byte, FO, signifies a break code. The third byte is the same scan code as Key 44, the left-Shift key, but the EO prefix told you it's not really that key. The last two bytes are the unshifted ↑ make code.

When the right Shift (Key 57, scan code 59) is down, you get EO FO 59 EO 75. The keyboard sends an ersatz break key for the shift key and then the usual make code.

The ↑ break code becomes EO FO 75 EO 12 when the left Shift is down. The controller sends the normal ↑ break code (EO FO 75) and then reshifts with EO 12. The right-Shifted break code is EO FO 75 EO 59, of course.

We're not done yet!

The system keyboard controller translates these keyboard scan codes into somewhat less formidable system

now from
DON LANCASTER

ACTIVE FILTER COOKBOOK
The **sixteenth (!)** printing of **Don's bible** on **analog op-amp lowpass**, bandpass. and **highpass active filters. De-mystified instant designs. $28.50**

CMOS AND TTL COOKBOOKS
Millions of copies in print worldwide. THE two books for **digital integrated circuit fundamentals About as hands-on as you can get. $24.50 each.**

INCREDIBLE SECRET MONEY MACHINE II
Updated 2nd edition of Don's classic on setting up your own technical or craft venture. $18.50

LANCASTER CLASSICS LIBRARY
Don's best early stuff at a **bargain** price. **Includes** the CMOS Cookbook, The TTL Cookbook, **Active** Filter Cookbook, PostScript video, Case Against **Patents, Incredible Secret** Money Machine II, and Hardware Hacker II **reprints. $119.50**

LOTS OF OTHER GOODIES

Ask the Guru I or II or III	$24.50
Hardware Hacker II or III	$24.50
Micro Cookbook I	$19.50
PostScript Beginner Stuff	$29.50
PostScript Show and Tell	$29.50
Intro to PostScript Video	$29.50
PostScript Reference II	$31.50
PostScript Tutorial/Cookbook	$19.50
PostScript by Example	$31.50
Understanding PS Programming	$29.50
PostScript: A Visual Approach	$22.50
PostScript Program Design	$ 24.50
Thinking in PostScript	$ 22.50
LaserWriter Reference	$19.50
Type 1 Font Format	$15.50
Acrobat Reference	$24.50
Whole works (all PostScript)	$380.00
PostScript Insider Secrets	FREE
Hacking Insider Secrets	FREE

POSTSCRIPT SECRETS
A Book/Disk combination crammed full of **free fonts**, insider resources. utilities, **publications, workarounds**, fontgrabbing more. **For** most any **PostScript** printer. Mac or PC format **$29.50**

BOOK-ON-DEMAND PUB KIT
Ongoing details on Book-on-demand publishing, a new method of producing books only when and as ordered. Reprints, sources. **samples. $39.50**

THE CASE AGAINST PATENTS
For most, individuals, patents are virtually **certain** to result in a net loss of sanity, energy. **time, and** money. This two volume set shows you **tested** and proven real-world alternatives. $28.50

BLATANT OPPORTUNIST I
The reprints from all Don's Midnight Engineering columns. Includes a broad range of real **world** proven **coverage** on small scale technical **startup** ventures. Stuff you can use right now. **$24.50**

RESOURCE BIN I
A complete collection of all Don's Nuts **& Volts** columns to date, including a new **index** and **hi6** master names and numbers list **$24.50**

FREE SAMPLES
Well. nearly free anyway. Almost Do join **us on** GEnie PSRT to sample all of the Guru 6 **goodies.** The downloading cost on a typical Guru IS is 21 cents. Modem access: (800)638-8369, then a JOINGENIE. Use DMD524 for your **keycode.**

FREE VOICE HELPLINE **VISA/MC**

SYNERGETICS
Box 809-CC
Thatcher, AZ 85552
(520) 428-4073

#124

scan codes. Even though the new make codes can be two bytes long, the 8042 converts them to single-byte values, possibly with an EO prefix for the new keys. The unshifted ↑ make code becomes EO 48 and the break code is EO C8. With the left Shift down, you get EO AA EO 48 and EO C8 EO 2A. You can probably guess that the system scan code for left Shift is 2A. Similarly, with the right Shift down, you get EO B6 EO 48 and EO C8 EO 36.

If you understand that, what happens if both shift keys are down when you press ↑? ‘Obviously, the make code is EO AA EO B6 EO 48 and the break code is EO C8 EO 36 EO 2A. What could be easier?

OK, final question: what does left-Shift-PrtSc produce? Would you believe EO 2A EO 37 EO B7 EO AA? That’s in addition to the 2A and AA produced by the left Shift key, of course. The PrtSc key is a make-only key, which means you won’t get a break code when it goes up. That explains, at least a little bit, why the make code includes the break code.

The BBS files this month display the system keyboard-controller output for each key. The IRQ 1 interrupt handler shown in Listing 1 reads the system scan codes from the controller and places them in a ring buffer. A display routine running in Demo Task 3 extracts the codes and writes them on the VGA display. If you have a scope or logic analyzer, you can also watch the keyboard scan codes on the cable and compare them with the system scan codes.

You should also compare the system scan codes with those in your references. So far, every table I’ve seen has errors, omissions, or misinterpretations. Some are simple typos (watch those Dees, Ohs, Zeros, Eyes, Ells, and Ones!), while others are more serious. Spend a while tapping the keys, reading the books, and making marginal notes.. . .

Don’t forget that there is yet *another* translation layer. All those old application programs called the original BIOS keyboard functions and should not be offended by unexpected key codes. When the BIOS reads a “new” system scan code from the

Listing 2—This table gives the real-mode BIOS return values for each of the Enhanced keyboard’s 101 keys. The comments show the key’s Scan Code Set 3 number in hex, the physical key number in decimal, and the keycap legend. Scan codes 13 and 53 correspond to keys found only on non-U.S. 102-key keyboards. Those keyboards lack the key that produces scan code 5C. The keyboard interface described next month will use this table to generate BIOS-compatible values.

```
          STRUC     KEYCODES
BaseCase  DW        0                           ; unshifted character
CapCase   DW        0                           ; Caps shift
CtrlCase  DW        0                           ; Ctrl shift
AltCase   DW        0                           ; Alt shift
          ENDS      KEYCODES

          LABEL     KeyCodes BYTE
           --- Standard PC codes ----            Sc Key  Keycap
,          Base Caps Ctrl Alt                   ; Cd Num Legend
 KEYCODES <>                                    ; 00
 KEYCODES <>                                    ; 01
 KEYCODES <>                                    ; 02
 KEYCODES <>                                    ; 03
 KEYCODES <>                                    ; 04
 KEYCODES <>                                    ; 05
 KEYCODES <>                                    ; 06
 KEYCODES <03B00h,05400h,05E00h,06800h>         ; 07112 F1
 KEYCODES <0011Bh,0011Bh,0011Bh,00100h>         ; 08110 Escape
 KEYCODES <>                                    ; 09
 KEYCODES <>                                    ; OA
 KEYCODES <>                                    ; OB
 KEYCODES <>                                    ; OC
 KEYCODES <00F09h,00F00h,09400h,0A500h>         ; OD  16 Tab
 KEYCODES <02960h,0297Eh,00000h,02900h>         ; OE   1 `~
 KEYCODES <03C00h,05500h,05F00h,06900h>         ; OF113 F2
 KEYCODES <>                                    ; 10
 KEYCODES <00000h,00000h,00000h,00000h>         ; 11  58 Left Ctrl
 KEYCODES <00000h,00000h,00000h,00000h>         ; 12  44 Left Shift
 KEYCODES <>                                    ; 13  45 102-key only
 KEYCODES <00000h,00000h,00000h,00000h>         ; 14  30 Caps Lock
 KEYCODES <01071h,01051h,01011h,01000h>         ; 15  17 Q
 KEYCODES <00231h,00221h,00000h,07800h>         ; 16   2 1!
 KEYCODES <03D00h,05600h,06000h,06A00h>         ; 17114 F3
 KEYCODES <>                                    ; 18
 KEYCODES <00000h,00000h,00000h,00000h>         ; 19  60 Left Alt
 KEYCODES <02C7Ah,02C5Ah,02C1Ah,02C00h>         ; 1A  46 Z
 KEYCODES <01F73h,01F53h,01F13h,01F00h>         ; 1B  32 S
 KEYCODES <01E61h,01E41h,01E01h,01E00h>         ; 1C  31 A
 KEYCODES <01177h,01157h,01117h,01100h>         ; 1D  18 W
 KEYCODES <00332h,00340h,00300h,07900h>         ; 1E   3 2@
 KEYCODES <03E00h,05700h,06100h,06B00h>         ; 1F 115 F4
 KEY CODES <>                                   ; 20
 KEYCODES <02E63h,02E43h,02E03h,02E00h>         ; 21  48 C
 KEYCODES <02D78h,02D58h,02D18h,02D00h>         ; 22  47 X
 KEYCODES <02064h,02044h,02004h,02000h>         ; 23  33 D
 KEYCODES <01265h,01245h,01205h,01200h>         ; 24  19 E
 KEYCODES <00534h,00524h,00000h,07B00h>         ; 25   5 4$
 KEYCODES <00433h,00423h,00000h,07A00h>         ; 26   4 3#
 KEYCODES <03F00h,05800h,06200h,06C00h>         ; 27116 FS
 KEYCODES <>                                    ; 28
 KEYCODES <03920h,03920h,03920h,03920h>         ; 29  61 Space
 KEYCODES <02F76h,02F56h,02F16h,02F00h>         ; 2A  49 V
 KEYCODES <02166h,02146h,02106h,02100h>         ; 2B  34 F
 KEYCODES <01474h,01454h,01414h,01400h>         ; 2C  21 T
 KEYCODES <01372h,01352h,01312h,01300h>         ; 2D  20 R
 KEYCODES <00635h,00625h,00000h,07C00h>         ; 2E   6 5%
 KEYCODES <04000h,05900h,06300h,06D00h>         ; 2F 117 F6
 KEYCODES <>                                    ; 30
 KEYCODES <0316Eh,0314Eh,0310Eh,03100h>         ; 31  51 N
 KEYCODES <03062h,03042h,03002h,03000h>         ; 32  50 B
 KEYCODES <02368h,02348h,02308h,02300h>         ; 33  36 H
 KEYCODES <02267h,02247h,02207h,02200h>         ; 34  35 G
```

(continued)

Listing 2-continued

```
KEYCODES <01579h,01559h,01519h,01500h> ; 35  22  Y
KEYCODES <00736h,0075Eh,0071Eh,07D00h> ; 36   7  6^
KEYCODES <04100h,05A00h,06400h,06E00h> ; 37 118  F7
KEYCODES <>                            ; 38
KEYCODES <00000h,00000h,00000h,00000h> ; 39  62  Right Alt
KEYCODES <0326Dh,0324Dh,0320Dh,03200h> ; 3A  52  M
KEYCODES <0246Ah,0244Ah,0240Ah,02400h> ; 3B  37  J
KEYCODES <01675h,01655h,01615h,01600h> ; 3C  23  U
KEYCODES <00837h,00826h,00000h,07E00h> ; 3D   8  7&
KEYCODES <00938h,0092Ah,00000h,07F00h> ; 3E   9  8*
KEYCODES <04200h,05B00h,06500h,06F00h>; 3F 119  F8
KEYCODES <>                            ; 40
KEYCODES <0332Ch,0333Ch,00000h,03300h> ; 41  53  ,<
KEYCODES <0256Bh,0254Bh,0250Bh,02500h> ; 42  38  K
KEYCODES <01769h,01749h,01709h,01700h> ; 43  24  I
KEYCODES <0186Fh,0184Fh,0180Fh,01800h> ; 44  25  0
KEYCODES <00B30h,00B29h,00000h,08100h> ; 45  11  0)
KEYCODES <00A39h,00A28h,00000h,08000h> ; 46  10  9(
KEYCODES <04300h,05C00h,06600h,07000h>;  47120  F9
KEYCODES <>                            ; 48
KEYCODES <0342Eh,0342Eh,00000h,03400h> ; 49  54  .>
KEYCODES <0352Fh,0353Fh,00000h,03500h> ; 4A  55  /?
KEYCODES <0266Ch,0264Ch,0260Ch,02600h> ; 4B  39  L
KEYCODES <0273Bh,0273Ah,00000h,02700h> ; 4C  40  ;:
KEYCODES <01970h,01950h,01910h,01900h> ; 4D  26  P
KEYCODES <00C2Dh,00C5Fh,00C1Fh,08200h> ; 4E  12  -_
KEYCODES <04400h,05D00h,06700h,07100h>; 4F 121  F10
KEYCODES <>                            ; 50
KEYCODES <>                            ; 51
```

(continued)

8042, it returns the best match among the "old" codes. For example, unshifted ↑ returns the old 8-↑ AH/AL values: 48/00. Keys that didn't exist on the old keyboard and have no valid equivalents are simply discarded.

New programs, however, can call a different BIOS entry point to get the new extended codes: unshifted ↑ returns 48/E0, while 8-↑ returns the familiar 48/00. This is the level nearly all PC programmers work at-with the mysteries of keyboard and system scan codes carefully hidden from view.

In protected mode, we can't take advantage of the BIOS functions. Given what you know now, are you up to writing a BIOS keyboard replacement?

Me, neither.

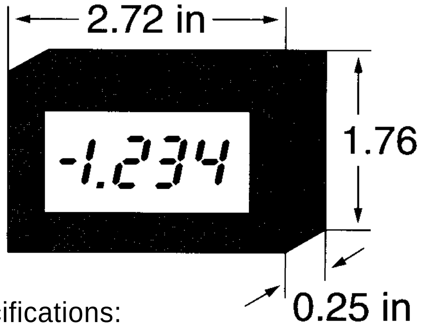
SCANNING THE CODES

Here's where a bit of history comes in handy. Recall that IBM designed the Enhanced keyboard for all its systems. While the convoluted scan codes might be necessary for backwards PC compatibility, they made

3½-DIGIT LCD PANEL METER

-Available now at an unheard of price of $15 plus s& h

New! Not surplus!



Specifications:

Maximum input: k199.9 mV additional ranges provided through external resistor dividers

Display: 3½-digit LCD, 0.5 in. figure height, jumper-selectable decimal point

Conversion: Dual slope conversion, 2-3 readings per sec.

Input Impedance: >100M ohm

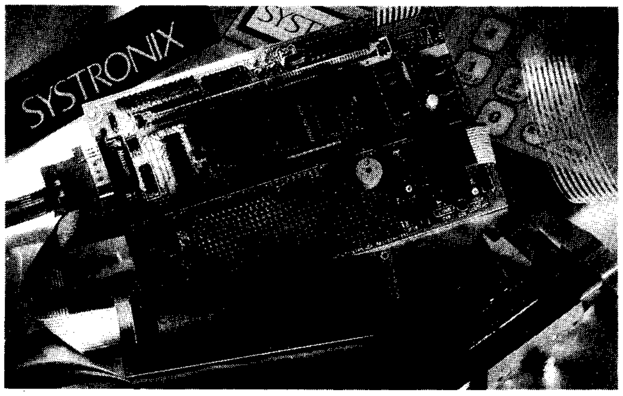
Power: 9-12 VDC @ 1 mA DC

Circuit Cellar, Inc.

4 Park Street, Suite 12, Vernon, CT 06066

Tel: (203) 875-2751 Fax: (203) 872-2204

NEW! UNIVERSAL DALLAS DEVELOPMENT SYSTEM from **$199!**



- It's a complete **8051-family** single board computer!
- One board accommodates any 40 DIP **DS5000, 40** SIMM **DS2250, 40** SIMM DS2252, or 72 SIMM **DS2251,** 8051 **superset** processor! Snap one out, snap another in.
- Programs via PC serial port. Program lock & encrypt.
- LCD interface, keypad decoder, **RS232** serial port, 8-bit ADC, four relay driver outputs, four buffered inputs.
- Power with 5VDC regulated or 6-13 VDC unregulated
- Large prototyping area, processor pins routed to headers
- Optional enclosures, keypads, **LCDs,** everything you need
- BCI51 Pro BASIC Compiler w/50+ Dallas keywords $399

SYSTRONIX® TEL: 801.534.1017 **FAX: 801.534.1019**
555 South 300 East, Salt Lake City, UT, USA 84111

#125

little sense for any other application. Can you imagine anyone voluntarily working backwards from the keyboard output to figure out which key was pressed?

The Enhanced keyboard microcontroller can encode keys using any one of three distinct scan-code sets. You've just seen a sample of the default, Scan Code Set 2, in all its glory. If you're writing a PC keyboard-controller BIOS, that's the one you must interpret to remain compatible with everyone else.

Scan Code Set 1 is closer to the original PC/AT 84-key encoding. Although the controller tracks the shift states and sends out ersatz shift keys as needed, the codes are entirely different than Set 2. I suspect this might have been a first pass at integrating the additional keys before resolving all the backwards compatibility issues. In any case, it's now ensnared in the PC Compatibility Barnacles.

Scan Code Set 3, on the other hand, clearly shows its non-PC heritage. Unlike the two other sets, every key sends a single, one-byte make code regardless of the shift state. Break codes are two bytes long: FO followed by the key's make code. No muss, no fuss, easy to decode, and easy to use-1 like it a lot.

Also, unlike Scan Code Sets 1 and 2, many of the keys are not typematic. For example, the Ctrl and Alt keys to the right of the space bar are make only. The controller sends a single make code regardless of how long the key stays down and does not send a break code when it goes up.

Using Scan Code Set 3 in a real-mode PC application isn't an option, but in protected mode the Compatibility Barnacles don't bind us quite so tightly. Listing 2 should give you an idea of what's ahead. It lists the real-mode BIOS return values for each key alone and then with Caps-, Ctrl-, and Alt-shift. The keys are in Scan Code Set 3 order rather than the usual Set 2 hodgepodge.

Next month, we'll build a BIOS-compatible protected-mode keyboard interface without chewing through all those incomprehensible scan codes. Not only am I sure you've never seen this trick before, but I bet you never knew your keyboard had more than one scan-code set. Right?

Meanwhile, you can experiment with the keyboard in real mode using good old Debug. Set up a DOS boot diskette with the following AUTOEXEC.BAT file:

```
mode com1:9600,n,8,1
ctty com1
```

Then, fire up the a comm program on your main PC. Boot the '386SX system into DOS and type DOS and Debug commands through the serial link!

You must disable the BIOS keyboard handler before sending controller commands and reading keyboard scan codes. The easiest way is masking IRQ 1 at the 8259 interrupt controller. Read port 21 hex to get the current IMR, OR that value with 02

Listing 2—*continued*

```
KEYCODES <02827h,02822h,00000h,02800h> ; 52 41  '"
KEYCODES <>                            ; 53 52  102-key only
KEYCODES <01A5Bh,01A7Bh,01A1Bh,01A00h> ; 54 27  [{
KEYCODES <00D3Dh,00D2Bh,00000h,08300h> ; 55 13  =+
KEYCODES <08500h,08700h,08900h,08B00h> ; 56122  F11
KEYCODES <00000h,00000h,07200h,00000h> ; 57 124 Print Screen
KEYCODES <00000h,00000h,00000h,00000h> ; 58 64  Right Ctrl
KEYCODES <00000h,00000h,00000h,00000h> ; 59 57  Right Shift
KEYCODES <01C0Dh,01C0Dh,01C0Ah,01C00h> ; 5A 43  Enter
KEYCODES <01B5Dh,01B7Dh,01B1Dh,01B00h> ; 5B 28  ]}
KEYCODES <02B5Ch,02B7Ch,02B1Ch,02B00h> ; 5C 29  \| 101-key only
KEYCODES <>                            ; 5D
KEYCODES <08600h,08800h,08A00h,08C00h> ; 5E 123 F12
KEYCODES <00000h,00000h,00000h,00000h> ; 5F 125 Scroll Lock
KEYCODES <050E0h,050E0h,091E0h,0A000h> ; 60 84  Gray Down
KEYCODES <04BE0h,04BE0h,073E0h,09B00h> ; 61 79  Gray Left
KEYCODES <00000h,00000h,00000h,00000h> ; 62126  Pause
KEYCODES <048E0h,048E0h,08DE0h,09800h> ; 63 83  Gray Up
KEYCODES <053E0h,053E0h,093E0h,0A300h> ; 64 76  Gray Del
KEYCODES <04FE0h,04FE0h,075E0h,09F00h> ; 65 81  Gray End
KEYCODES <00E08h,00E08h,00E7Fh,00E00h> ; 66 15  Backspace
KEYCODES <052E0h,052E0h,092E0h,0A200h> ; 67 75  Gray Ins
KEYCODES <>                            ; 68
KEYCODES <04F00h,04F31h,07500h,00000h> ; 69 93  1 End
KEYCODES <04DE0h,04DE0h,074E0h,09D00h> ; 6A 89  Gray Right
KEYCODES <04B00h,04B34h,07300h,00000h> ; 6B 92  4 Left
KEYCODES <04700h,04737h,07700h,00000h> ; 6C 91  7 Home
KEYCODES <051E0h,051E0h,076E0h,0A100h> ; 6D 86  Gray PgDn
KEYCODES <047E0h,047E0h,077E0h,09700h> ; 6E 80  Gray Home
KEYCODES <049E0h,049E0h,084E0h,09900h> ; 6F 85  Gray PgUp
KEYCODES <05200h,05230h,09200h,00000h> ; 70 99  0 Insert
KEYCODES <05300h,0532Eh,09300h,00000h> ; 71104  . Del
KEYCODES <05000h,05032h,09100h,00000h> ; 72 98  2 Down
KEYCODES <04C00h,04C35h,08F00h,00000h> ; 73 97  5 (pad)
KEYCODES <04D00h,04D36h,07400h,00000h> ; 74102  6 Right
KEYCODES <04800h,04838h,08D00h,00000h> ; 75 96  8 Up
KEYCODES <00000h,00000h,00000h,00000h> ; 76 90  Num Lock
KEYCODES <0E02Fh,0E02Fh,09500h,0A400h> ; 77 95  Gray /
KEYCODES <>                            ; 78
KEYCODES <0E00Dh,0E00Dh,0E00Ah,0A600h> ; 79108  Gray Enter
KEYCODES <05100h,05133h,07600h,00000h> ; 7A 103 3 PgDn
KEYCODES <>                            ; 7B
KEYCODES <04E2Bh,04E2Bh,09000h,04E00h> ; 7C 108 Gray +
KEYCODES <04900h,04939h,08400h,00000h> ; 70101  9 PgUp
KEYCODES <0372Ah,0372Ah,09600h,03700h> ; 7E 100 Gray *
KEYCODES <>                            ; 7F
KEYCODES <>                            ; 80
KEYCODES <>                            ; 81
KEYCODES <>                            ; 82
KEYCODES <>                            ; 83
KEYCODES <04A2Dh,04A2Dh,08E00h,04A00h> ; 84 105 Gray
```

hex, and write it back. On my system, the IMR is normally B8, so a simple 0 21 BA masks the keyboard interrupt. Obviously, you must be running Debug through the serial port when you disable the keyboard interrupt!

Check the references for the system board keyboard controller's I/O ports and bit definitions. I'll cover these in detail next month. A little advance exploration on your part will make things more comprehensible.

RELEASE NOTES

Demo Taskette 3 installs a keyboard hardware interrupt handler and displays standard system scan codes on the VGA display. Try all the Ctrl, Alt, and Shift combinations for the "gray keys" while you're at it. Be amazed at the number of scan codes for PrtSc, Break, and similar keys.

The task dispatcher now displays the number of task switches per second on the VGA's bottom line. At 33 MHz, the '386SX clocks about 400 switches per second, executing each taskette 100 times each second.

Although these numbers take on more significance next month, I'd be interested to hear the results on your system.

If you didn't pick up Frank Van Gilluwe's ***The Undocumented PC*** (Addison Wesley, ISBN o-201-62277-7) last month, it's too late by now. Your fellow readers got there first. It has the best rendition of the keyboard interface I've ever seen, and other chapters are equally good. There are a few typos in the keyboard table, of course.

The tables in Hogan's ***Programmer's PC Sourcebook*** (Microsoft Press, ISBN 1-55615-321-X) summarize the various keyboards and scan code sets. The keyboard controller commands are not documented, and there are no references to the PS/2 system keyboard controller functions.

Thanks to Rick Freeman and the folks at Computer Options in Raleigh for helping me check out my ideas. I left at 6:01 Saturday evening with one each of every keyboard they had and returned at 9:59 Monday. It was quite a Sunday!

Next month, we dive back into heavy-duty PM coding. Without this month's background, it won't make any sense at all!

Ed Nisley, as Nisley Micro Engineering, makes small computers do amazing things. He's also a member of Circuit Cellar INK's engineering staff. You may reach him at *ed.nisley@circellar.com* ***or*** 74065.13638 *compuserve.com.*

SOFTWARE

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

IRS

413 **Very Useful**
414 Moderately Useful
415 Not Useful

HCS II
Home Control System

● Energy **Management** ● Security **and** Alarm ● **Coordinated** Home Theater ● **Coordinated** Lighting ● **Monitoring** and Data Collection ●

Get all these capabilities and more with the Circuit Cellar HCS II. Call, write, or FAX us **for a brochure. Available as-**sembled or as a kit.

Circuit Cellar HCS

CIRCUIT CELLAR, INC.
4 Park **Street, Suite 12** • Vernon, CT **06066**
Tel: (203) 875-2751 • Fax: (203) 872-2204

#126

Emulating a Motorola IR Chip Using a PIC

FROM THE BENCH

Jeff Bachiochi

When chips and support products disappear, what do you do? According to Jeff, you engineer your way around the problem. Here, Jeff uses a PIC to emulate the original Motorola infrared transmission codes to get back in business.

OK, I admit it. I was wrong. I believed 8-track tapes would outlive cassettes. I wasn't even close. I guess what I liked the best-not having to rewind-was not enough for everybody.

My cartridge machine now collects dust along with my TRS-80 Model I. Both have been recycled into mass-storage devices.

Sad to say, you can't always turn an orphaned piece of equipment into a useful member of society. Many times, as with early Polaroid Land cameras, no matter how long the equipment continues to be operational, the loss of a primary ingredient (e.g., the film) renders it useless.

And so it is with the IR-Link, first introduced in *INK 26*. Steve covered infrared tracking and remote control and Ed divulged his software secrets on producing and recognizing IR transmissions. The IR-Link board based IR recognition on the Manchester-encoded 9-bit transmission scheme used by Motorola's MC145030. Rather than use the chip, the IR-Link encodes and decodes in software.

To send IR commands to the IR-Link, it is necessary to train a hand-held trainable remote with the IR-Link's codes. That remote can then be used as an input device to the Home Control System (HCS).

Enhanced and upgraded, the IR-Link became the MCIR-Link. Additional features enabled the MCIR-Link to be trained to reproduce (not recognize) most audio/video remote transmissions. With this, the HCS gained the ability to control remote equipment like a TV, VCR, and so on.

THREE YEARS

It has been about three years since the introduction of those first HCS network modules. We've finally come to the time that we have to make adjustments for a manufacturer discontinuing a particular part.

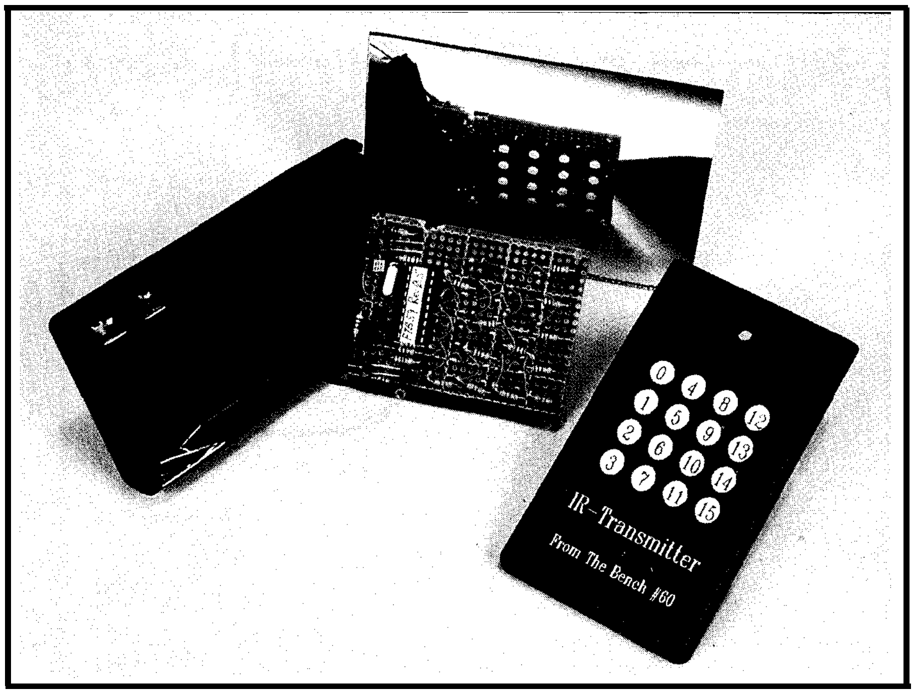


Photo 1—*A PIC-based hand-held IR transmitter eliminates the need for a trainable remote. It replicates the codes necessary for the MCIR-Link. As you can see, the prototype has components mounted on both sides of the protoboard.*

Motorola has chosen to halt production of the MC145030. Although this does not directly affect the MCIR-Link or the HCS, its loss has become a thorn in the side. On its own, this handicap goes unnoticed. But, with Radio Shack also discontinuing its only trainable IR remote, we have a problem. It was the only trainable IR-remote universally available to MCIR-Link users.

And now, without the Motorola MC145030, users can't even prototype a hand-held remote using the original device. We can't blame Radio Shack—trainable remotes are unnecessary with today's preprogrammed all-in-one units.

It's just too bad these preprogrammed units don't support the '030's 9-bit Manchester transmission codes!

A ROSE BY ANY OTHER NAME

This article is for new users who want to use the MCIR-Link as an IR-input control to the HCS. Without a trainable remote or at least the MC145030 to prototype a hand-held remote, we're gonna have some unhappy campers!

But wait. Microchip is tenting next door! Luckily, camp neighbors are some of the most friendly people on this earth. A PIC16C54 helps solve this dilemma and puts us back on the trail of total home control.

PIC devices are a perfect match for situations requiring battery power, keypad scanning, and direct drive of LEDs. Take a look at Figure 1 and Photo 1. The simplicity of the circuit makes a strong statement to design engineers. Notice no power switch is necessary. Power is always on. The trick here is the PIC's sleep mode. In sleep mode, the oscillator is halted, creating a static condition requiring only a few microamps (in some cases less than 1 µA).

A reset pulse on the *MCLR input wakes the processor as if power was applied. As shown in Microchip's embedded control handbook, a wake-up on key press is accomplished by setting a logic low on all columns prior to going to sleep. The controller holds the last output levels during sleep.

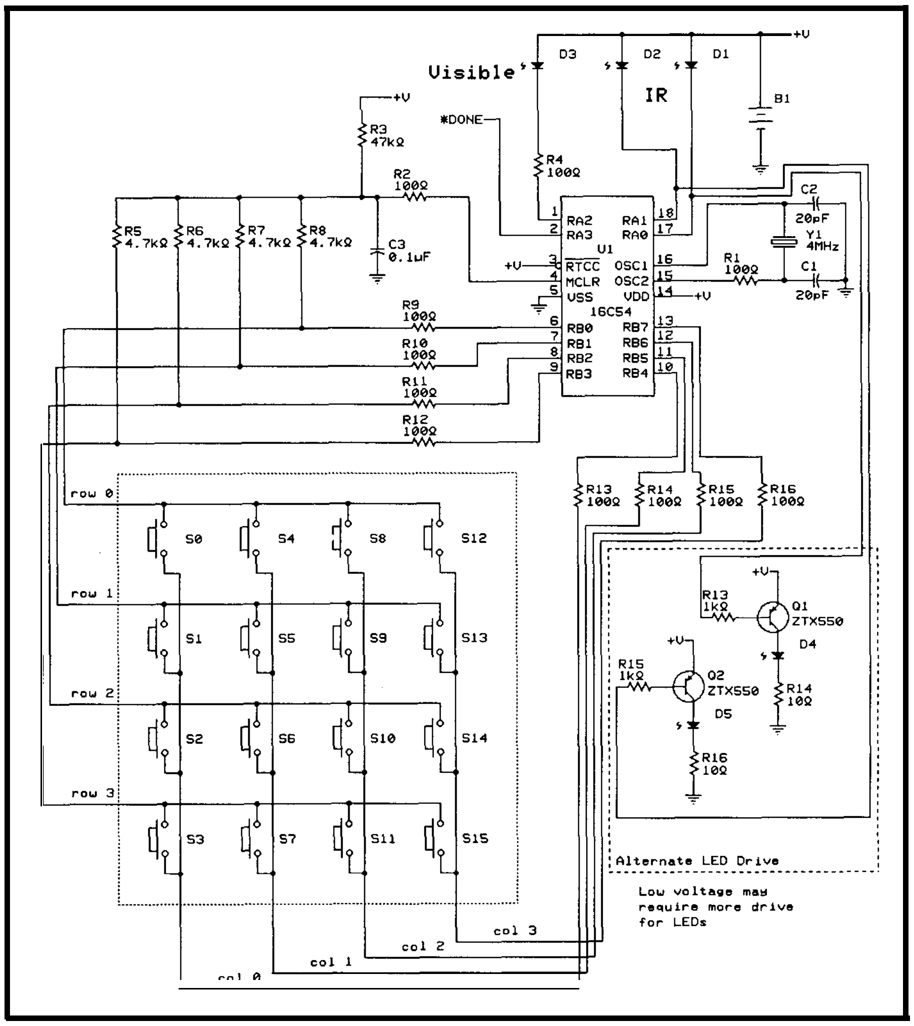


Figure 1 —*The PIC processor in the battery-operated IR transmission controller handles all aspects of the unit's operation including keypad scanning and decoding, IR transmission timing, and putting itself to sleep.*

Any key pressed after the processor has gone to sleep discharges the capacitor holding up the * MCLR input. Once discharged past the reset threshold, the processor goes into reset.

The I/O pins are tristated during reset, removing the ground applied to • MCLR through the pressed key, and the reset sequence continues (on *MCLR's rising edge). About 18 ms after 'MCLR goes high, code execution can begin. Notably, the processor is able to wake up and go about its business even if the pressed key (which started the activity) is still down.

You'll notice later on that the key scan routine (which also grounds the columns) does not pull down *MCLR. This is due to the timing of the column scan pulse. No reset occurs if the grounded column does not exceed the discharge time of the RC on • MCLR's input.

AUTOMATIC TRANSMISSION

To mimic the MC145030 transmissions as recognized by the MCIR-Link, we need to adhere to two timings:

1) the transmission bit time (the time to transmit 1 bit of data)
2) the modulation rate or the frequency of the IR during the actual transmission of data

The bit time is fixed at 1290 µs. This is split into two 645-µs time periods. The complement of the data is sent during the first time period and the data itself during the second. Therefore, the only two legal bit transmissions are a logic low followed by a logic high (data = 1) or a logic high followed by a logic low (data = 0). Although the 1290-µs bit time may start out at either logic level, the logic level changes 645 µs into the bit time.

Because I'm generating the modulation frequency and bit timing from the same oscillator, the two must be in sync and resolution is limited to I µs (based on the 4-MHz crystal). During the 645 µs of IR transmission, we use a modulation frequency close to 38 kHz (the center frequency of the infrared receiver module on the MCIR-Link) and evenly divisible by 645 µs.

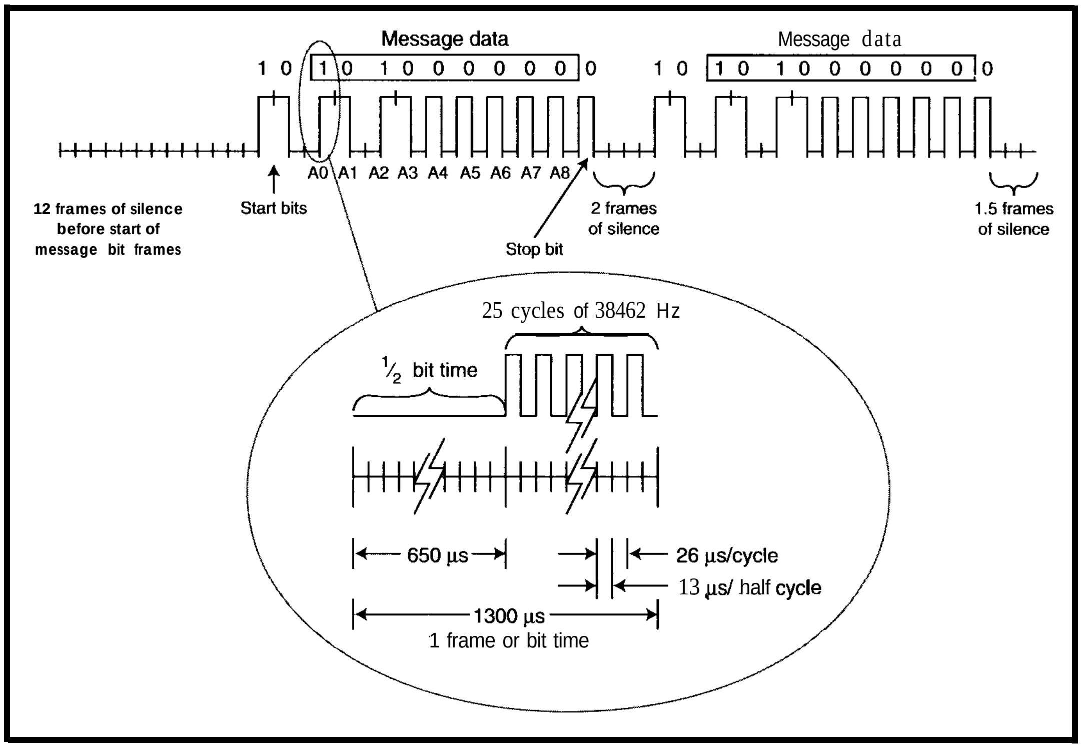


Figure **2**—*The transmission timing shows the Manchester encoding of data. Each message is repeated twice. The actual timing varies slightly from Motorola's specs, but the deviance is less than 1%.*

To illustrate how I reached this value, let's work through the math. First we have to determine the number of cycles:

$$\#\text{of cycles} = \frac{1}{2}\text{ bit time} \times \text{mod freq}$$
$$= 0.000645 \times 38000$$
$$= 24.51$$

If we round up the number of cycles to 25, it is easy to find the modulation frequency we end up with:

$$\text{Freq} = \frac{\#\text{ of cycles}}{\frac{1}{2}\text{ bit time}}$$
$$= \frac{25}{0.000645}$$
$$= 38760$$

I chose to use 25 cycles of 38760 Hz to represent each logic-high transmission. Actually, I use 38462 Hz since it's the nearest whole microsecond period possible with 1-µs instruction times (i.e., $\frac{1}{38759}$ = 25.8 ps, so $\frac{1}{26}$ µs = 38462).

Refer to Figure 2 for a typical transmission burst. Notice the transmission is made up of a quiet period (12 bit times) followed by a start bit (actually a 1 data bit followed by a 0 data bit), 9 data bits, and a stop bit (a 0 data bit). After two quiet bit times, the start, data, and stop bits are repeated a second time, ending with a 1.5-bit-time quiet period. This produces a total of

$$12 + 2 + 9 + 1 + 2 + 2 + 9 + 1 + 1.5 = 39.5 \text{ bit times}$$

Although the actual IR is only transmitted for a maximum of 26 bit times or 34 ms, the quiet times allow the receiver to recover from one transmission and prepare for the next.

512 POSSIBILITIES

The tiniest PIC (16C54) has 12 I/O lines. I use three of these to directly drive the IR and visible LEDs (the PIC doesn't require external drivers). One more line is necessary for the *DONE output. This left me with eight.

I contemplated using these with rocker/slide switches and stealing back a bit as a send-code input. But, I just couldn't see the user remembering what the 256 switch settings were. I also pondered having the user press a three-digit code, but buried that quickly for the same reasons. The 16-key keypad, while not providing access to all 5 12 possibilities, has the advantage of being user friendly.

THE SOFTER SIDE

The PIC source code is not complicated. Figure 3a describes the program's main section. On reset, the temporary registers, Option register, and ports are initialized.

The Option register sets up the RTCC using a divide-by-8 prescaler. Each tick of the RTCC is 8 ps. I use the RTCC for long delays by allowing an inner loop to count up to 7Dh for an inner loop time of 1 ms (125 x 8 µs = 1 ms). This loop is accurate to within a few microseconds because of peculiar timing and coding situations.

Port A (a four-bit port) is set up as all outputs. Three of the outputs drive the LEDs (one visible and two IR) and the fourth is used as a debugging signal. It presently signals a logic low prior to going to sleep (*DONE).

Port B (an eight-bit port) is split as half inputs and half outputs. The lower nybble acts as row inputs while the upper nybble functions as column driver outputs.

After a brief (50-ms) switch-debouncing delay (long delay), the scan routine is called to detect which key has been pressed. On return, if no key has been pushed (as may happen when power is applied for the first time), the

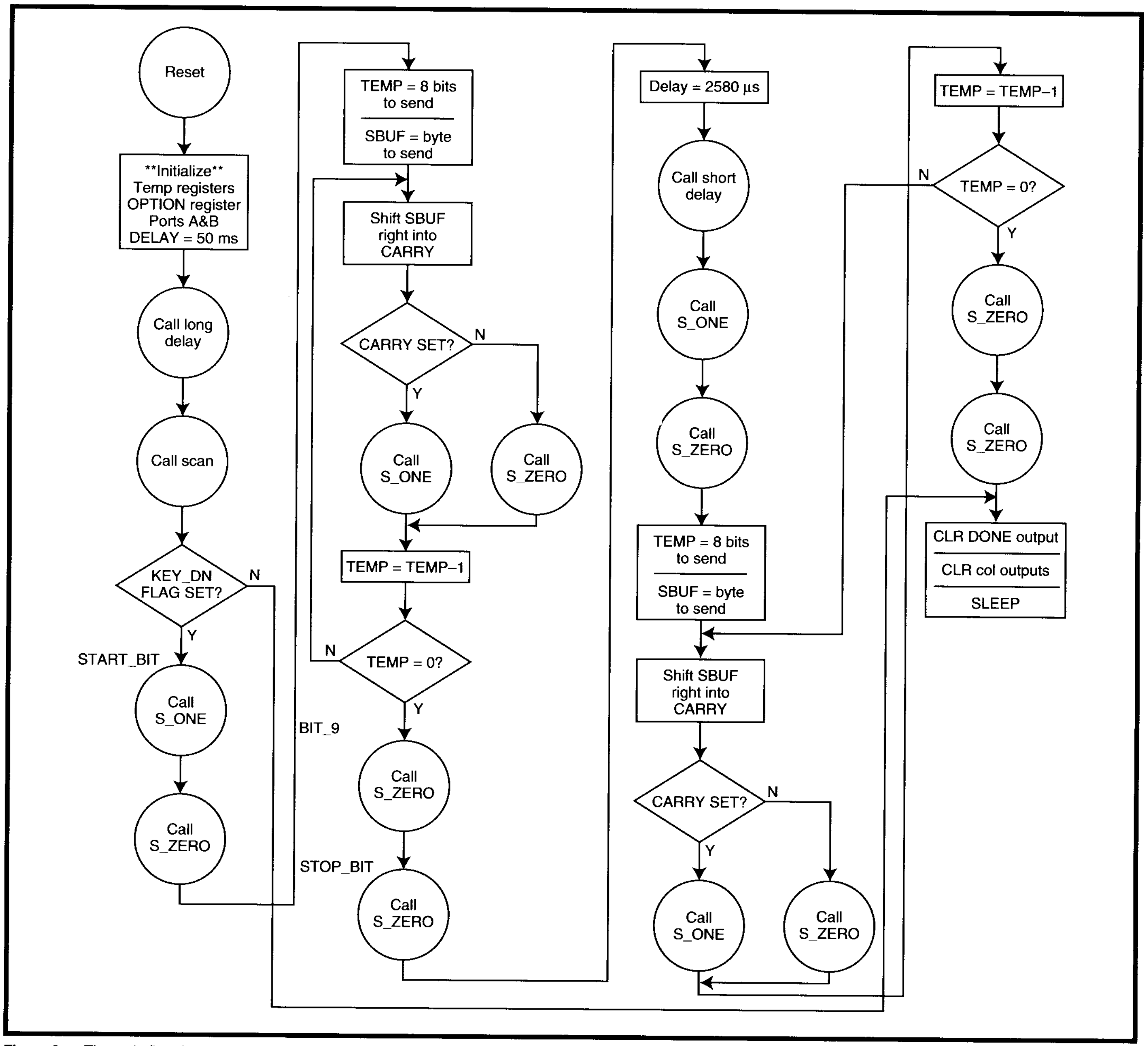


Figure 3a—*The main flowchart shows the steps between a Reset and the power down into Sleep mode.*

row outputs of Port B are set to logic 0 and the SLEEP instruction is executed.

The PIC shuts off its oscillator and draws very little current. Any key press shorts out the capacitor, holds *MCLR high, and initiates a reset, thus beginning the cycle again.

4×4

The scan routine (the details aren't shown in the figure) enables a column by applying a logic zero, checks to see if any of the rows are pulled down by a key press, and disables the column.

Once a key press is found, the value associated with that key is stored in SBUF and the scan routine exits. The scan of each column is so short that a held-down key does not cause the *MCLR capacitor to discharge and restart the code as it does once the processor goes into Sleep mode. The only exception to this is when a key is held down until the IR transmission finishes and the code has put the PIC to sleep.

Once the scan routine returns a key value in SBUF, the IR transmission begins. The 12-bit-time silence is not needed here since we have already been quiet for at least 50 ms.

Two routines are used to transmit IR: S_ZERO and S_ONE. Each routine is 1 bit time long (i.e., 1300 µs). Each routine is split into the two 650-µs halves with each half basically doing the same exercise: do something or nothing, wait 13 µs, do something or nothing, wait 13 µs, and repeat it 25 times.

If the "somethings" alternately enable and disable the LEDs, then we produce 25 cycles of the LEDs flashing at 38462 Hz. If the "nothings" are NOPs, then we just rest for the specified time. The S_ZERO routine transmits IR the first half bit time and rests the second half while the S_ONE routine rests the first half and transmits IR during the second half.

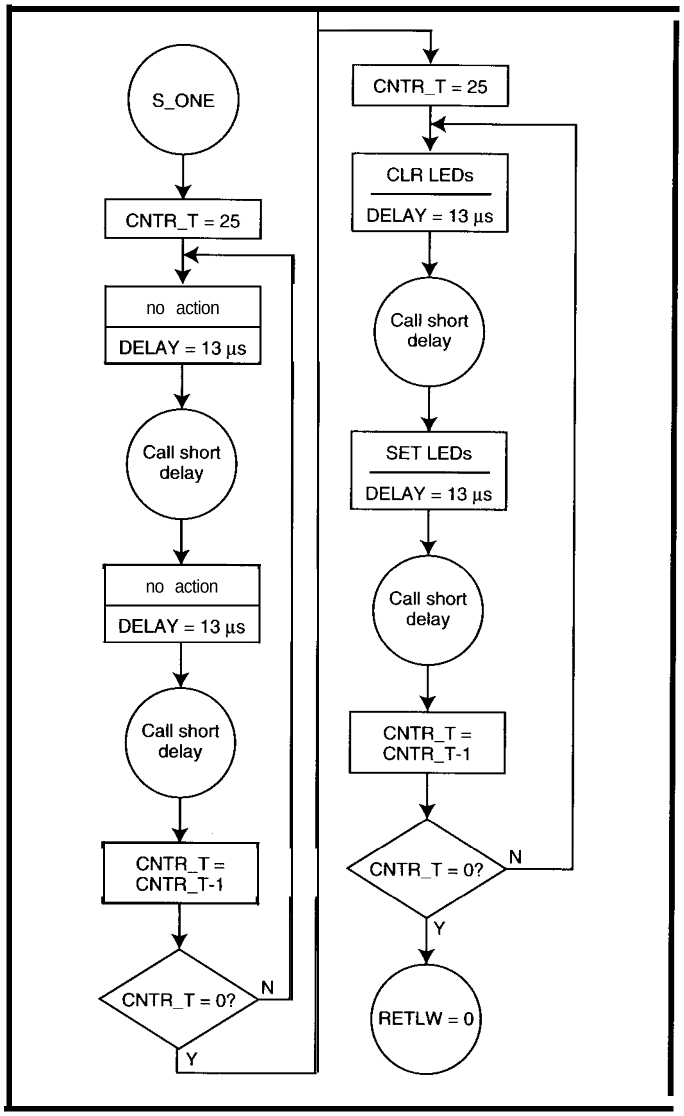


Figure **3b**—*Sending a 1 data bit involves no action for a period of time, followed by turning the LED off and back on again.*

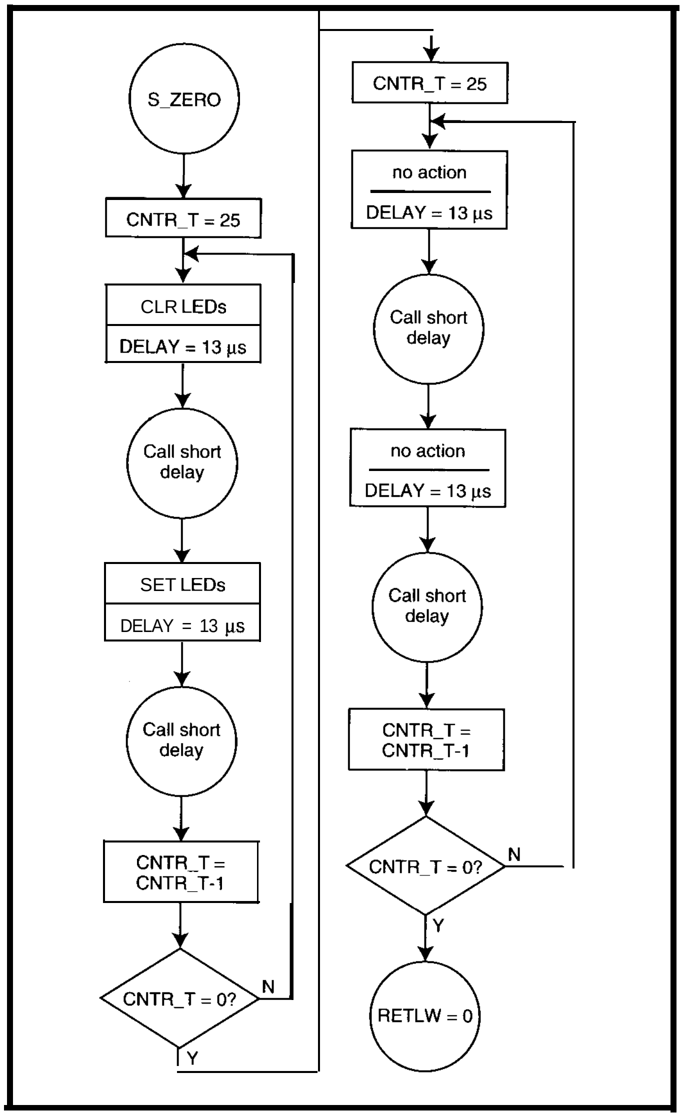


Figure **3c**—*Sending a zero bit is the same as sending a one* **bit**, *but the order of actions is reversed. The LED is cleared and set, then no action is taken for a period of time.*

To reproduce the Motorola MC145030's format, we merely call the appropriate S_ZERO and S_ONE routines (see Figures 3b and 3c) in the proper sequence. The start bit is a two-bit sequence of a 1 and 0. The actual data, in this case from SBUF values of O-15, comes next LSB to MSB.

Since the format calls for 9 bits of data, we simply append a 0 as the ninth bit (using a 1 would send codes equaling 256 + 0 through 256 + 15). Finally, the required stop bit [i.e., a data 0) is sent. Once this 13-bit word is completed, the entire sequence is repeated a second time after a short two bit-time pause.

Now that our transmission has ended, *DONE and the column outputs are set to logic 0 in anticipation of another wake-up call. The processor again executes the SLEEP instruction.

STILL AWAKE?

I hope you're still with me.

After all, I consider this to be a perfect example of how flexible the PIC can be in solving potentially irritating problems. It makes a handy little circuit for experimenting with IR transmissions. Those of you with PIC development kits (even just the assembler and programmer) will find that the same basic circuit covers most transmission schemes. Ya' can't get much simpler.

Jeff ***Bachiochi (pronounced "BAH-key-AH-key") is an electrical engineer on Circuit Cellar INK's engineering staff. His background includes product design and manufacturing. He may be reached at*** *jeff.bachiochi@circellar.corn.*

SOURCE

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

Preprogrammed PIC16C54—
FTB59-1 ....$10

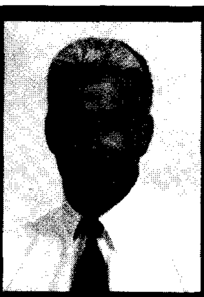
I R S

416 Very Useful
417 Moderately Useful
418 Not Useful

Chip On Patrol

SILICON UPDATE

Tom Cantrell



You want an 8-bit controller that's not an 8051, but is still cost-effective and can provide a solid midrange solution? Well, have you tried National's COP family? According to Tom, COP's the way to go.

If you're getting harassing letters and phone calls from a crazed stalker, it's time to call a cop.

On the other hand, if you're the Joe or Jane design engineer and the threats are coming from your boss, maybe it's time to call a COP-as in the so-named embedded micro lineup from National Semiconductor.

I suspect I'm not alone in admitting that National isn't the first company that comes to mind when considering a micro. Motorola, Intel, Philips, Zilog, Microchip sure, but National? Though well respected for linear products (regulators, amps, A/D converters, etc.), it's fair to say that National's got more than a few micro skeletons (PACE, 32000, Swordfish) rattling around in their closet.

Still, the COP chips have had some success, though you may not know it since the parts are deeply buried in consumer and automotive applications. However, I hadn't realized the breadth of their general-purpose 8-bit lineup, nor what seems to be a new enthusiastic commitment to the general-purpose microcontroller market.

There are those who cry, "Who needs another micro?" But as the embedded market blows past one billion units per year, I'd like to point out that more, not less, specialization is both likely and desirable.

It's a mistake-I call it "micro-myopia"-to bury your head in the silicon and arbitrarily dismiss or ignore would-be contenders. In fact, sometimes parts that are initially overlooked come back to life in a big way (e.g., the PIC and ARM).

Therfore, I suggest you put aside your preconceptions [or your lack of any conceptions) about the COP8 family and make sure you give it a fair trial.

JUST THE FACTS, MA'AM

The fact is the COP8 family covers a broad spectrum of low- to mid-range single-chip applications, thanks to myriad permutations of memory size and type (i.e., ROM, EPROM, OTP), temperature range (commercial: 0–70°C, industrial: –40–+85°C, and military: –55–+125°C), packaging (DIP, PLCC, and SO), and I/O features (Photo 1).

At the entry level, the '822C offers a minimalist 1 -KB EPROM and 64

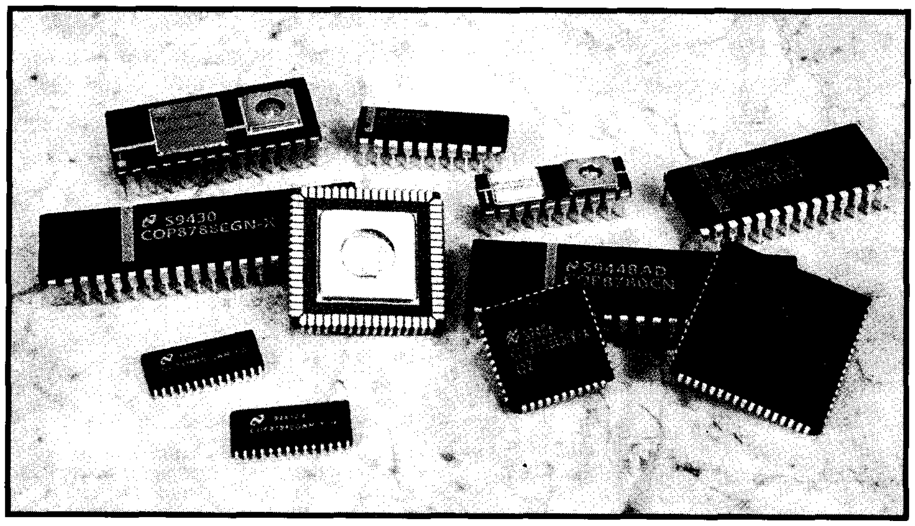


Photo 1—*Talk* about *COPs on the street! Literally dozens of versions of the COP blanket a wide range in price, performance, and applications. Note the unique "hybrid" (two die) EPROM* offerings.

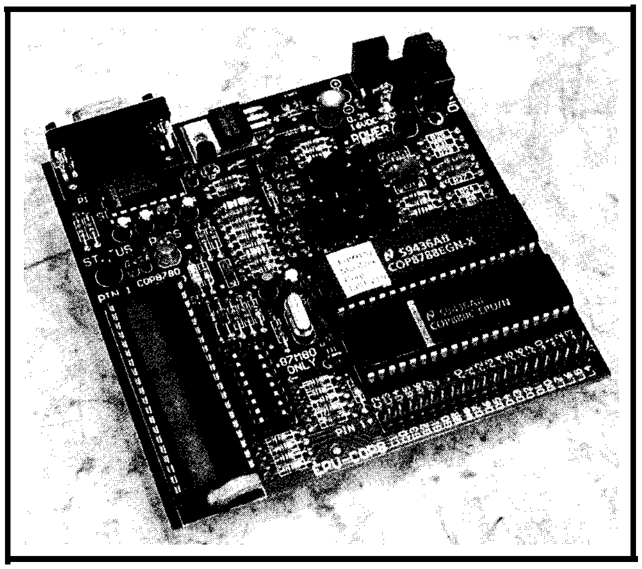


Photo **2**—*The EPU* (Evaluation and Programming Unit) isn't real-time, *but* at *only $125, it's a real steal.*

bytes of RAM with 16 I/O lines packed into a space-saving 20-pin package (DIP or SO). At less than $4 in hundreds for OTP ($6 for a windowed part), it's certainly competitive with more well-known penny-pinching OTPs. Those with the cash, commitment, and courage to go for a masked ROM take note of the '912C which, by making a few sacrifices (768 bytes ROM versus 1 -KB EPROM, commercial versus industrial temperature range, 2-µs versus l-us instruction cycle), breaks the $1 barrier in high volume.

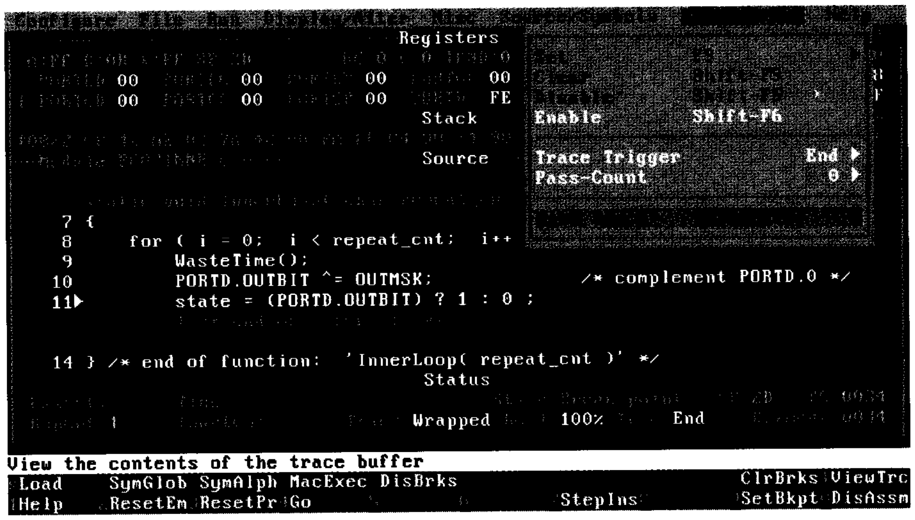
At the other end of the spectrum, the new 888EK matches the most modern feature-laden micros with 8 KB code memory, 5 12 bytes RAM, and 36 I/O lines in a 40-pin DIP (or 44-pin PLCC). Notably unique features include special reduced-EM1 techniques and an interesting analog subsystem with on-chip multiplexer, comparator, and constant-current source. At under $10 for OTP (and less than $5 for ROM), the 888EK delivers a lot of bits (memory and I/O) for your bucks.

When it comes to waging your own personal war on bugs, National offers something for everyone with a range of tools from a full-featured, real-time in-circuit emulator to a low-cost evaluation and programmer board. One nice feature is that all the tools come from one supplier (Metalink), so there's only one set of commands to learn whether you're using a $100 EV board or a $10,000 emulator.

For the cost conscious, the entry-level tools may prove surprisingly effective. Consider the EPU-COP8 (Evaluation and Programming Unit pictured in Photo 2). Designed to specifically support the 4-KB OTP, 128-byte data RAM, 40/44-pin '880C, it can also be used to develop code for smaller pin count and memory members of the family, but not burn their on-chip EPROM.

Sure, the EPU doesn't handle full-speed, real-time debugging, but what do you expect for a miserly $1252 Despite the low price, the EPU gets remarkable mileage (breakpoints, passpoints, trace, etc.) and includes the COP8 assembler and linker to boot.

The EPU also comes with a demo version of the ByteCraft C compiler, which is fully compatible with the debugger. Photo 3 shows screen shots of the setup in action. Notice the support for source-level debug (both ASM, C, or mixed)-impressive for such a low-cost package. Admittedly,


```
                                 Registers
         00          00          00          00      Enable          Shift-F6
         00          00          00          FE
                                     Stack           Trace Trigger        End ▶
                                                     Pass-Count             0 ▶
                                     Source
  7 {
  8     for ( i = 0;  i < repeat_cnt;  i++
  9         WasteTime();
 10         PORTD.OUTBIT ^= OUTMSK;                 /* complement PORTD.0 */
 11▶        state = (PORTD.OUTBIT) ? 1 : 0 ;

 14 } /* end of function:  'InnerLoop( repeat_cnt )' */
                                     Status
                                 Wrapped        100%        End
View the contents of the trace buffer
 Load     SymGlob SymAlph MacExec DisBrks                          ClrBrks ViewTrc
 Help     ResetEm ResetPr Go                       StepIns         SetBkpt DisAssm
```

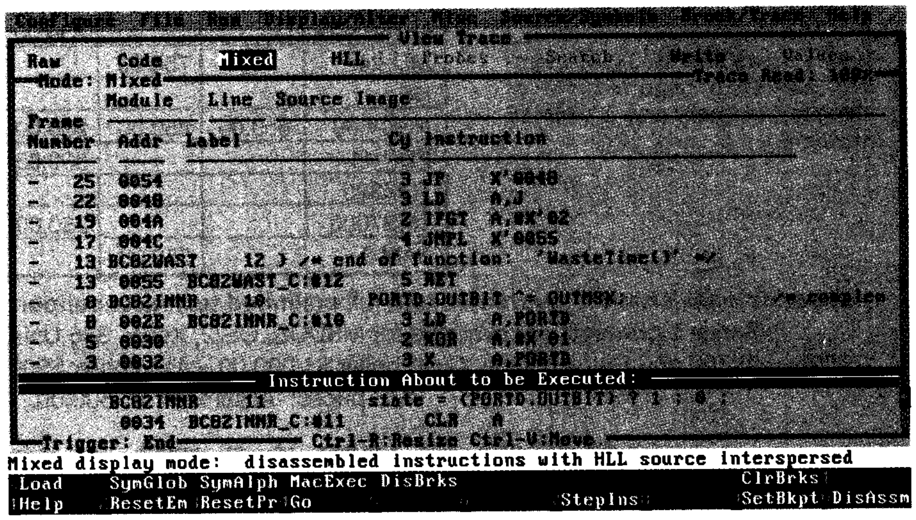

```
                              View Trace
 Raw      Code      Mixed      HLL
 Mode: Mixed                                               Trace Read: 100%
          Module    Line  Source Image
 Frame
 Number   Addr  Label                    Cy Instruction

 -    25  0054                           3 JP     X'0048
 -    22  0048                           3 LD     A,J
 -    19  004A                           2 IFGT   A,#X'02
 -    17  004C                           4 JMPL   X'0055
 -    13 BCB2WAST      12 } /* end of function:  'WasteTime()' */
 -    13  0055  BCB2WAST_C:#12           5 RET
 -     8 BCB2INNR      10          PORTD.OUTBIT ^= OUTMSK;           /* comple
 -     8  002E  BCB2INNR_C:#10           3 LD     A,PORTD
 -     5  0030                           2 XOR    A,#X'01
 -     3  0032                           3 X      A,PORTD
 ───────────────── Instruction About to be Executed: ─────────────────
         BCB2INNR      11          state = (PORTD.OUTBIT) ? 1 : 0 ;
          0034  BCB2INNR_C:#11             CLR    A
 Trigger: End ─────────── Ctrl-R:Resize Ctrl-V:Move ──────────────
Mixed display mode:  disassembled instructions with HLL source interspersed
 Load     SymGlob SymAlph MacExec DisBrks                          ClrBrks
 Help     ResetEm ResetPr Go                       StepIns         SetBkpt DisAssm
```

Photo *3-The EPU* debugger screen (top) shows a simple C demonstration program that toggles a bit (bit *0* of Port *D). Once* the breakpoint is hit (note how *Port D* in the register window has gone *from FFh to FEh), a trace buffer* showing prior execution can *be viewed (bottom).*

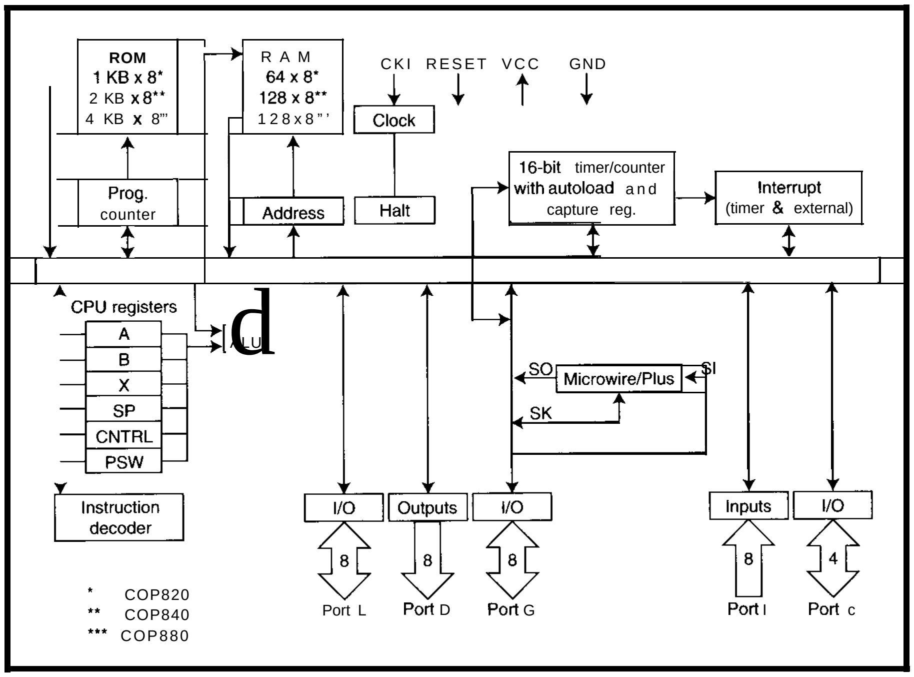


Figure l--Even *the simplest COPs* include a fairly powerful timer and a *Microwire/Plus* serial bus

the limited code and data space of the chips discourages HLL bloatware, but a few hundred lines of simple bit-banging integer-only C code should fit without problem.

GOING UNDERCOVER

Whether a 16-pin jellybean or 68-pin smorgaschip, all COP8s feature a compatible architecture and certain standard function modules. Fortunately, it's quite straightforward and refreshingly simple (see Figure 1).

Most of the general-purpose port pins are bidirectional and features a choice of modes (Hi-Z input, input with weak pull-up, or push-pull output). Port D is further distinguished by high-current (10 mA) outputs suitable for driving transistors or LEDs.

Most of Port G is distinguished by optional special functions (G1 and G2 remain general-purpose). G7 is called into play as CKO to act as the second pin (along with the dedicated CKI pin) for connecting a crystal. Alternatively, an external clock or RC can be connected to CKI only, in which case G7 is available as a general-purpose input or a restart input that takes the chip out of low-power Halt mode. The clock input is 10 times the CPU speed (i.e., the maximum spec of 10 MHz corresponds to a l-us instruction cycle). Speed is somewhat limited (i.e., 2-3 MHz) for the RC option.

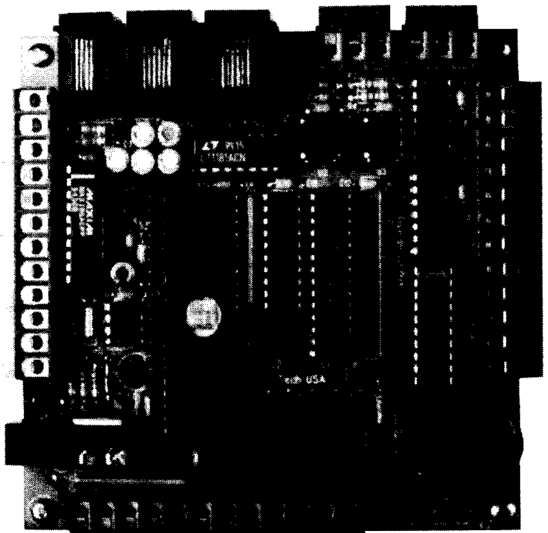
The least-significant bit of Port G (GO) can be configured to act as an external interrupt (INT) input with

It's time again for the NOW 7th Annual Circuit Cellar Design Contest

The time is now for you to start thinking about your entry.
Just write or call Rose at:
Circuit Cellar Design Contest
4 Park Street, Suite 90
Vernon, CT 06066
(203) 875-2199 or fax (203) 872-2204.
Entries must be in by September 15, 1995

Mid-Tech USA

Very high performance C-programmable controllers and development software



The ec.52 803 l-compatible 8 MIPS controller
Call for full-line hardware/software product catalog

Mid-Tech Computing **Devices** USA
P.O. Box 2 l8 • Stafford, CT 06075 • voice/fax: (203) 684-2442

#130

programmable polarity. On the entry-level chips, INT joins the on-chip timer and software I NT R instruction to make a total of three possible interrupt sources. Higher-end COPs offer many more interrupt sources, both external and internal.

G3 (TIO) performs double duty as the on-chip 16-bit timer (Figure 2) input and output pin. The timer can be configured for multiple modes: periodic interrupt, PWM, external counter, and input capture. The timer on older COPs runs at the instruction-cycle rate (i.e., 1 us) while some of the newer units run at the clock rate (i.e., 100 ns}, which is quite speedy indeed.

G4, G5, and G6 collectively function as a clocked-serial [i.e., shift register) interface known as Microwire/Plus. Like other shift register schemes (e.g., Motorola SPI and Philips PC), the Microwire/Plus port provides a lean serial bus for connecting to other chips (e.g., serial EEPROMs, ADCs, display drivers, other COPs, etc.) as shown in Figure 3.

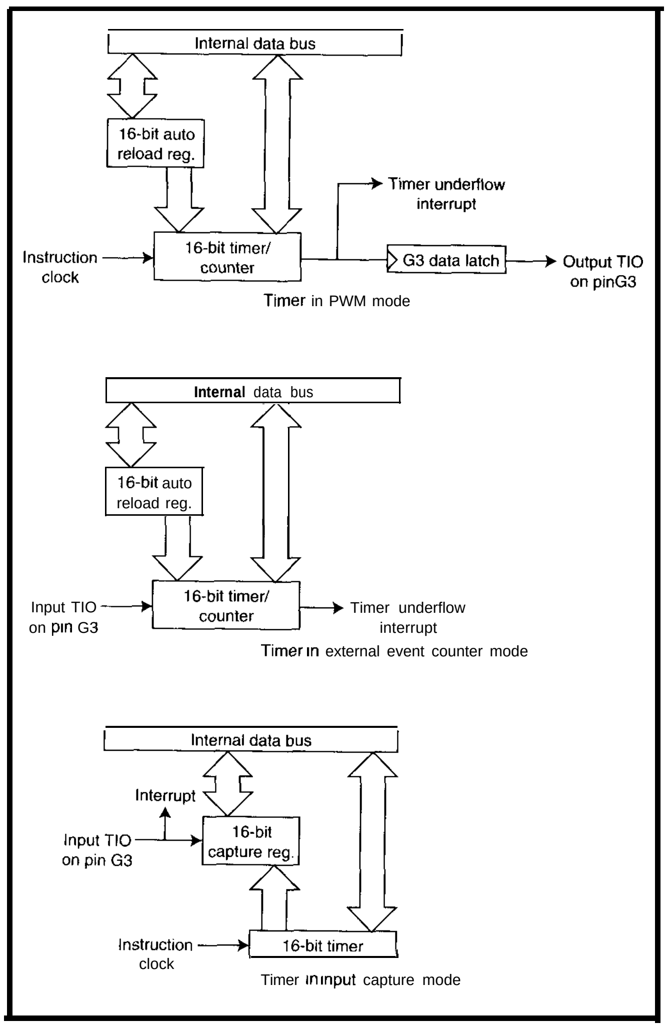


Figure 2—*The 16-bit* timer features *1-µs* resolution and a variety of input and output *modes via the TIO pin.*

Microwire/Plus uses a master/slave protocol in which one device (the master) is responsible for driving the shift clock (SK), which is then used by all other (slave) devices. Similarly, the master's output line (SO) is an input to slaves, and its input (SI) is sent by the slaves. As the figure shows, each slave also needs a dedicated chip-select line. Despite its humble pretensions, Microwire/Plus can deliver up to 500 kbps, which is more than enough bandwidth for a few low-speed I/O add-ons.

SPECIAL OPCODES AND TACTICS

While architectural wars continue to rage at the high end (i.e., 32 bits and beyond), for the class of applications the COP8 fits, too much technical hooey is much ado about nothing.

To make a long story short, the CPU seems like what you might end up with if you stuck all the other 8-bitters ('51, Z8, 'HC05, PIC, etc.) in a blender and punched frappe. Frankly, for the vast majority of middle-of-the-road applications, the architectural differences aren't really compelling.

Nevertheless, given the trend toward specialization, it's worth examining the COP's architecture with an eye toward particular strengths or limits.

One is that the COPs, unlike some others, were, are, and always will be single chip only. There's no provision for external bus expansion. Though the COP architecture conceivably supports up to 32 KB of code (i.e., 15-bit PC), it only supports 256 bytes of data. In fact, as Table 1 shows, it really only supports about 128 bytes of general-purpose data with the other 128 bytes allocated to on-chip I/O and control registers. Versions of the COP with more than 128 bytes of RAM rely on bank-select bits.

The 256-byte perspective yields a streamlined instruction set as shown in Table 2. Basically, a programmer deals with an accumulator (A), two index registers (B and X), and a stack pointer (SP)-all 8 bits. By today's criteria, the instruction set is rather baroque, but not more than other popular competitors.

One interesting feature is the reliance on skipping instructions to direct program flow instead of the conditional branches you may be used to. For instance, a COP subroutine

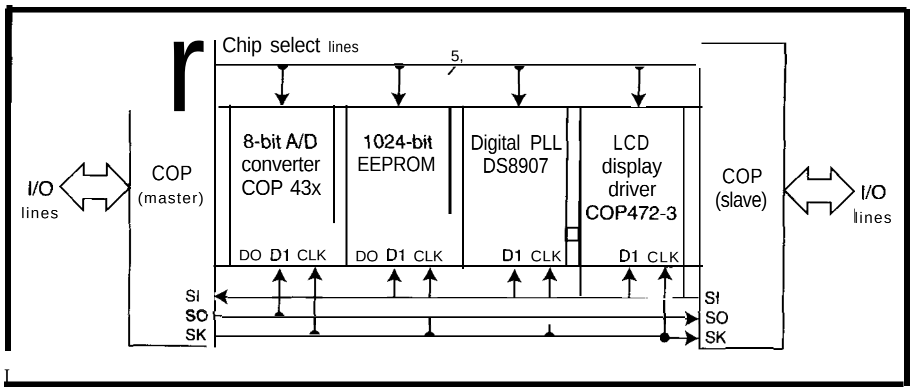


Figure 3—*Microwire/Plus, designed for moderate speed and minimum wiring I/O add-ons, is National's entry into the "LAN In A Box" fray,*

may pass a result to the caller by using a RET (return] or RETSK (return and skip next instruction) instead of trying to muck around with flags or stack info. The same goes for the I F instructions (e.g., I FEQ, I FGT, I FBN E) that evaluate the condition and perform or skip the next instruction depending on the evaluation result.

It's good that the COP doesn't have to hassle with flags or stack because it's barely got any. For example, there's no zero flag and the stack is really designed for calls and interrupts. Otherwise, the stack isn't very accessible (hint: rely on the fact the registers are mapped into the top 16 bytes of data space). Of course, 8-bit designers have learned to deal with similar challenges on other chips.

The COP features a good measure of bit handling, one of the most important jobs for this class of chips. Setting, resetting, or branching on any bit in the data space takes a single instruction. It can also perform BCD math (decimal correct instruction and half carry bit), a task that's otherwise rather cumbersome.

One welcome byproduct of the CISCy instruction set and 8-bit data address space is good code density. The COP may not be the fastest, but I suspect few micros can get the job done in fewer bytes.

When making a midrange controller decision, a particular system or I/O function may override petty architecture arguments. Scattered among the dozens of COP offerings are a number of features that may prove pivotal in a design.

Ironically, in this era when designers' eyes are bigger than their batteries, a figure of merit for a given micro is how well it does nothing (try that on your boss). COPs are sound sleepers, with Halt-mode consumption ranging from a few microamps to all the way down into nanoamps for the smallest parts. When halted, the chip (i.e., RAM and I/O held) can stay alive all the way down to 2 V.

Recognizing the down and dirty of consumer and industrial environments, certain COPs are ruggedized with quite a range of features including extended temperature range, watchdog timer, clock monitor, brownout protection, bad opcode and stack traps, less EM1 generation and susceptibility, and so on.

Besides the usual I/O suspects like UARTs and ADCs, you'll find some wise guys that might prove helpful in a dicey application. For instance, the previously mentioned mux and comparator analog block on the

Address	Contents
00-2F	48 on-chip RAM bytes
30–7F	Unused RAM address spaces (reads all 1 s)
00–6F	112 on-chip RAM bytes
70–7F	Unused RAM address spaces (reads all 1 s)
80–BF	Expansion space for on-chip EEPROM
CO-CF	Expansion space for I/O and registers
DO-DF	On-chip I/O and registers
D0	Port L data register
D1	Port L configuration register
D2	Port L input pins (read only)
D3	Reserved for Port L
D4	Port G data register
D5	Port G configuration register
D6	Port G input pins (read only)
D7	Port I input pins (read only)
D8	Port C data register
D9	Port C configuration register
DA	Port C input pins (read only)
DB	Reserved for Port C
DC	Port D data register
DD-DF	Reserved for Port D
EO-EF	On-chip functions and registers
EO-E7	Reserved for future parts
E8	Reserved
E9	Microwire/Plus shift register
EA	Timer lower byte
EB	Timer upper byte
EC	Timer autoload register lower byte
ED	Timer autoload register upper byte
EE	CNTRL control register
EF	PSW register
FO-FF	On-chip RAM mapped as registers
FC	X register
FD	SP register
FE	B register

Table 1—*The COP's 256-byte data space is split between general-purpose* RAM, I/O, and special function *(such* as *EEPROM* on certain models) registers. Note *the mapping of certain CPU registers (B, X, and SP) at the* fop of *the data address space.*

Tight Budget '51 Emulation

8051 Family Emulator is truly Low Cost!

The DryICE Plus is a modular emulator designed to get maximum flexibility and functionality for your hard earned dollar. The common base unit supports numerous 805 1 family processor pods that are low in price. Features include: Execute to breakpoint, Line-by-Line Assembler, Disassembler, SFR access, Fill, Set and Dump Internal or External RAM and Code, Dump Registers, and more. The DryICE Plus base unit is priced at a meager $299, and most pods run only an additional $149. Pods are available to support the 8031/2, 875 1/2, 80C154, 80C451, 80C535, 80C537, 80C550, 80C552/62, 80C652, 80C851, 80C320 and more. Interface through your serial port and a comm program. Call for a brochure or use INTERNET. We're at **info@hte.com** or ftp at ftp.hte.com

Tighter budget?
How about ***$149*** emulation?

Our $149 DryICE model is what you're looking for. Not an evaluation board - much more powerful. Same features as the DryICEPlus, but limited to just the 803 1/32 processor.

You can afford it!

So, if you're still doing the UV Waltz (Burn-2-3, Erase-2-3). or debugging through the limited window ROM emulators give, ***call us now*** for relief! Our customers say our products are still the best Performance/Price emulators available!

Look into our Single Board Computer solutions, too!

HTE
HiTech Equipment Corp
9400 Activity Road
San Diego, CA 92126
[Fax: (619) 530-1458]
Since 1983

(619) 566-1892

VISA MasterCard

Internet e-mail: info@hte.com
Internet ftp: ftp.hte.com

#131

ADD	add
ADC	add with carry
SUBC	subtract with carry
AND	logical AND
OR	logical OR
XOR	logical exclusive OR
IFEQ	IF equal
IFGT	IF greater than
IFBNE	IF B not equal
DRSZ	decrement register, skip if zero
SBIT	set bit
RBIT	reset bit
IFBIT	IF bit
X	exchange A with memory
LDA	load A with memory
LD mem	load direct memory immediate
LD Reg	load register memory immediate
X	exchange A with memory [B]
X	exchange A with memory [X]
LDA	load A with memory [B]
LDA	load A with memory [X]
LD M	load memory immediate
CLRA	clear A
INCA	increment A
DECA	decrement A
LAID	load A indirect from ROM
DCORA	decimal correct A
RRCA	rotate A right through C
SWAPA	swap nibbles of A
SC	set C
RC	reset C
IFC	IFC
IFNC	IF not C
JMPL	jump absolute long
JMP	jump absolute
JP	jump relative short
JSRL	jump subroutine long
JSR	jump subroutine
JID	jump indirect
RET	return from subroutine
RETSK	return and skip
RETI	return from interrupt
INTR	generate an interrupt
NOP	no operation

Table 2—*The COP instruction set is quite simple, though a little quirky. The good news is CISCy instructions make for high-code density.*

COP888 can implement a time-to-charge ADC scheme that provides maximum resolution of up to 14 bits at higher clock rates.

Other examples include a 16-bit PWM with 100-ns resolution, 64 bytes of EEPROM, hardware multiply and divide, and so on. There's even a COP, the '884BC, with a built-in CAN interface. This control-oriented LAN is finding acceptance in industrial and automotive applications (see Brad Hunting's articles in *INK 58* and 59).

TO PROSPECT AND SERVE

Getting support-questions answered, literature, samples, and so on-often seems a rather Catch-22 exercise. If you're a big customer [or lie well), you get support. Otherwise, you're shunted down the chain of command. You get the feeling you're interrupting someone with important things to do-like figuring out who to backstab at the next staff meeting. Finally, you end up in some voice-mail black hole that pleasantly vectors you back the way you came. Grrhh.

However, driven by competition, manufacturers seem to be catching on. "Gee, how can we get new customers if we only support old customers?"

National appears to have caught the "customers are good" bug. They're touting their Customer Support Center, an 800 number staffed from 7 A.M. to 7 P.M. (Central) with real live people who are supposedly prepared to help with technical questions, literature requests, or maybe even psychic readings if that's what it takes to get the chip designed in.

Too good to be true? Only way to tell is give 'em a call. Instead of dealing with a police-state mentality, maybe you'll find these COPs work for the Please Force.

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

CONTACT

National Semiconductor Corp.
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
(800) 272-9959 or (408) 721-5000
Fax: (800) 432-9672
BBS: (800) 672-6427 (8N1)

IRS

419 Very Useful
420 Moderately Useful
421 Not Useful

If you need I/O...

Add these numbers up:

80C552 a '51 Compatible Micro
40 Bits of Digital I/O
8 Channels of 10 Bit A/D
3 Serial Ports (RS-232 or 422/485)
2 Pulse Width Modulation Outputs
6 Capture/Compare Inputs
1 Real Time Clock
64K bytes Static RAM
1 + UVPROM Socket
512 bytes of Serial EEPROM
1 Watchdog
1 Power Fail Interrupt
1 On-Board Power Regulation

It adds up to real I/O power!

That's our popular **552SBC**, priced at just $299 in single quantities, Not enough I/O? There is an expansion bus, too! Too much I/O? We'll create a version just for your needs, and pass the savings on to you! Development is easy, using our Development Kit: The 552SBC-50 Development board with ROM Monitor, and an 8051 C compiler for just $449.

New Versions of the 8031SBC

Our popular 8031SBC can now be shipped with your favorite 8051 family processor. Models include **80C51FA, DS80C320, 80C550, 80C652, 80C154, 80C851** and more. Call for pricing today!

Truly Low-cost In-Circuit Emulator

The DryICE Plus is a low-cost alternative to conventional ICE products. Load, single step, interrogate, disasm, execute to breakpoint. Only $448 with a pod. For the 8051 family, including Philips and Siemens derivatives. Call for brochure!

Call for your custom product needs. Quick Response.

HTE
HiTech Equipment Corp.
9400 Activity Road
San Diego, CA 92126
(Fax: (619) 530-1458)
Since 1983

(619) 566-1892

VISA MasterCard

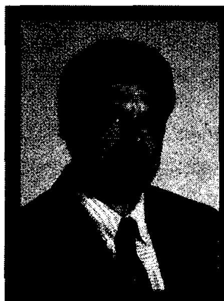
Internet e-mail: info@hte.com
Internet ftp: ftp.hte.com

#133

EMBEDDED TECHNIQUES

John Dybowski

How Small Can a Thermometer Get?



John uses an AT89C2051 micro-controller to network a DS1620 digital thermometer. A small LCD displays the temperature while RS-485 provides a network interface.

Combining highly integrated silicon with modern manufacturing results in smaller, more capable equipment. It pays to remember that the functionality presently crammed onto an area the size of a postage stamp occupied a whole room and required the support of a full facility not that long ago. Nowhere is miniaturization pushed to the extremes as it is in electronics.

Unfortunately, we often must produce prototypes and short production runs that cannot justify the up-front expense of a full surface-mount design. Things get even more difficult when such equipment must also satisfy the contradictory requirements of being very small and inexpensive.

By using the right parts regardless of the manufacturing techniques, we can enjoy enhanced silicon integration, which provides much denser functionality in our hardware. Obviously, for a certain class of products, this won't do. But in many cases, selecting the right parts keeps you in the running.

A couple of months ago, I featured the tiny (20-pin) flash-based Atmel AT89C205 1 8051 -compatible controller. I also presented a single-board design center that provides many of the capabilities of an in-circuit emulator and includes a built-in AT89C205 1 flash programmer.

Consider the alternative of programming an AT89C2051 every time you make a code tweak, and I'm sure you'll understand why I didn't attempt to design with the AT89C-205 1 until I had the development system operational. I've finally gotten my new system operating, so I hope you bear with me while I play with my new miniature micro. Photo 1 shows my new toys and the digital thermometer that is the subject of this column.

A THREE-CHIP THERMOMETER

I've been particularly interested in the development of very small embedded systems which operate as intelligent sensors or probes and can be located where the action is. Conversely, this tactic might result in keeping the central controller out of harm's way if the action occurs in an electrically hostile environment.

When such an instrument's functional parameters are defined to be sufficiently narrow, it is frequently referred to as a *smart sensor.* Although this classification may be unnecessarily restrictive, it has become common.

Although it extends the host controller's reach and capabilities, these small intelligent peripherals don't have to be slaves to a larger controller. Additional functions such as digital and analog I/O or memory components can be provided locally or remotely over a serial link. Standard multidropped RS-485 communications provide the most common and inexpensive means of linking such peripherals. A typical large-scale installation could involve a variety of high-performance controllers sharing a network link with small intelligent peripherals and subnets of smaller controllers slaved to the larger controllers at key locations.

With the cost of silicon dropping, it becomes easier to justify using processors in applications that tap only a fraction of their capabilities. In such cases, extremely fast project turnaround is possible by using a high-level language to render the application as I did with my digital thermometer.

In contrast, a hard-wired approach often proves to be simply unworkable, overly complicated, or prone to

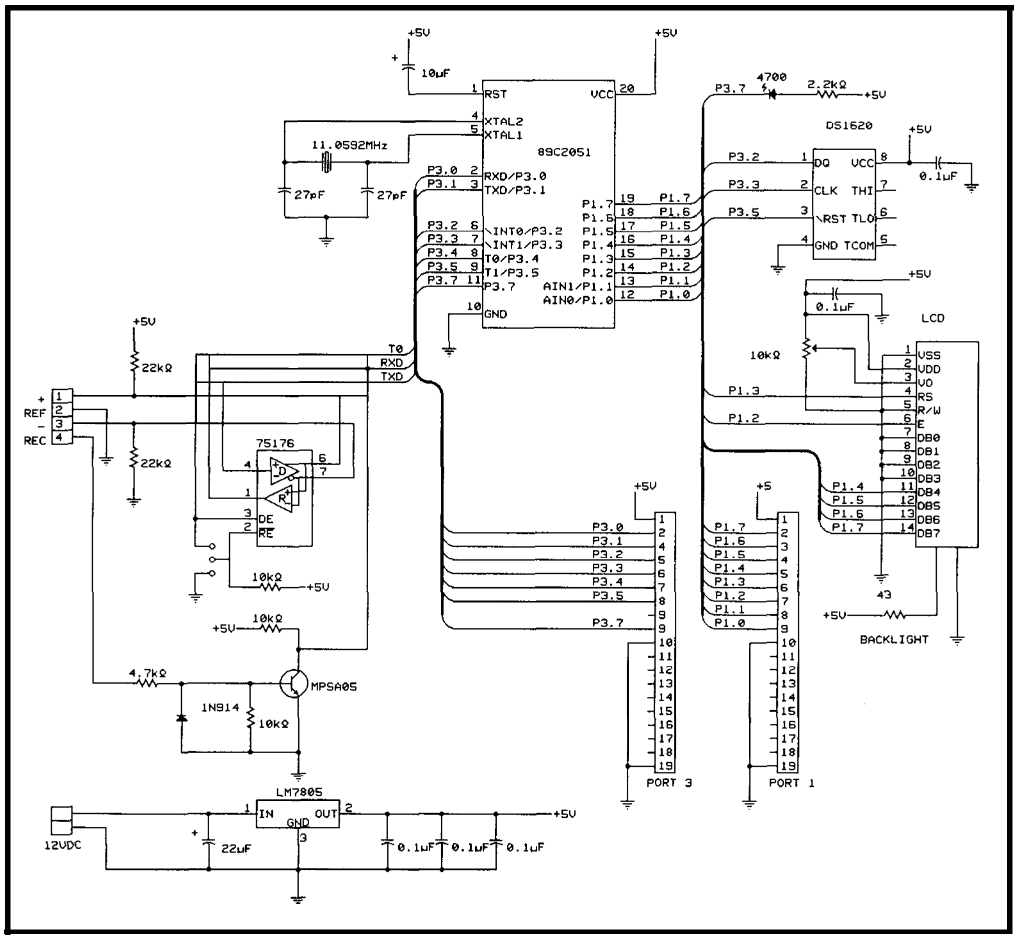


Figure I--By *faking full advantage of the microcontroller's port lines, necessary support logic can be kept to a minimum.*

system, shown schematically in Figure 1, is constructed on my AT89C205 1 protoboard and, as a result, requires minimal wiring.

excessive tweaking. Even if rendering your design purely in hardware is possible, the lack of flexibility ultimately proves unacceptable. Changing a few lines of code is much more attractive than making hardware modifications to a circuit card.

The system I am presenting this month consists of Atmel's AT89C2051 processor, a DS 1620 digital thermometer and thermostat from Dallas Semiconductor, an RS-485 and [quasi) RS-232 line interface, and a small 8 x 2 LED backlit LCD display. The small

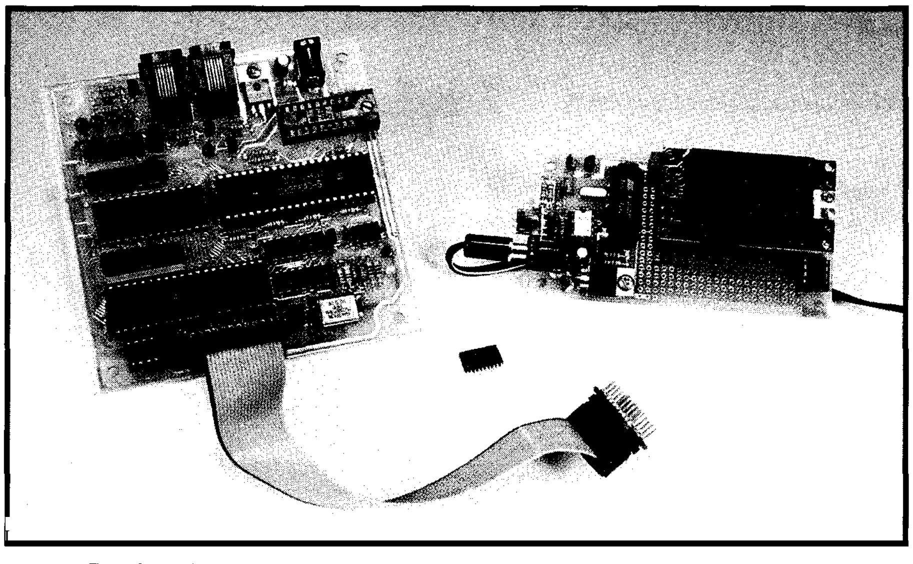


Photo 1—*The DS1620-based thermostat sports a backlit LCD display that can be read under any lighting* conditions. An A*T89C2051* development board aids in code debugging.

STANDARD DRIVERS

The firmware requirements for this small system are modest-initialization code, a main module, an LCD display driver, and a DS 1620 driver. The startup code is no problem since I am using Dunfield's Micro-C for my code generator. The run-time library provides everything I need to get from reset to the main module. For the display driver, I can choose from a repertoire of LCD support modules I've developed to run on 8051 processors. I have pure assembler drivers, C-callable assembler functions, and functions written entirely in C. These all support various LCD bus-width configurations including 8 bits, 4 bits, and 2-bit I^{2}C.

The most efficient code implementation depends on factors that are contradictory by nature. As you'd expect, assembly language drivers offer the ultimate in code efficiency and execution speed. On a controller with just 2 KB of program memory, there's strong incentive to use assembly language programming, but it boils down to what you're doing.

Of course, small processors have limitations beyond a tight program-storage area. Obviously, the number of I/O pins needed for the interface deserves careful consideration.

My ultimate hack in pin reduction involves an I^{2}C user I/O module I developed for another embedded controller project. This module is designed around PC-to-parallel converter chips and some supporting firmware routines. Using just two I/O pins, this module not only supports a 20 x 4 LCD, but also a 4 x 4 keypad, beeper, and leaves a few bidirectional I/O pins available for driving indicators or for other purposes.

Saving I/O pins, however, invariably requires more program memory and consumes more processor bandwidth since data and control sequences must be serialized and issued one bit at a time. Using the I^{2}C protocol also imposes additional protocol overhead and dictates a maximum clock rate of no more than 100 kHz. The slowdown, however, is not really an issue of display appearance since the display update appears to be instantaneous.

The LCD driver, shown in Listing 1, is written entirely in C and compiles under Micro-C using a tiny memory model. Although a stack-based implementation, Micro-C's special capabilities make it suitable for generating ROMable code for small systems. The additional overhead incurred performing stack manipulations is made up by the library functions that are all hand coded in highly optimized assembler.

Having recently completed a set of custom library extensions, I appreciate what Dunfield has accomplished in the overall scheme of his Micro-C. The

Listing 1--This *file contains support functions for the 8 x 2 LCD. The interface uses a direct* processor *I/O method in 4-bit mode.*

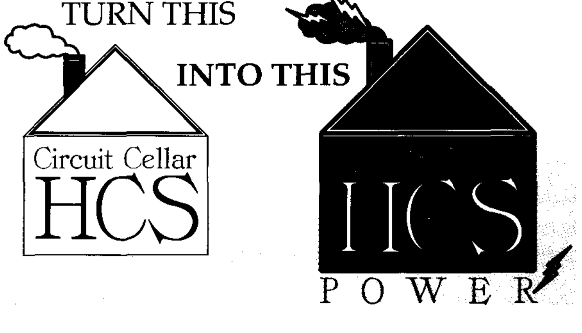
```
/* 8051 definitions  */
#include <8051reg.h>
#include <8051bit.h>

/* Defined bits for LCD control */
#define DEN P1.2
#define DRS P1.3

/* Global register (IRAM) data */
register unsigned char Cursor;

PutStr(unsigned char *p)      /* Putstr to LCD */
{
  unsigned char c;

  while (c = *p++)
    PutChar(c);
  return:
}

PutChar(unsigned char c)      /* Putchar to LCD */
{
  if (c == '\n'){
    if (Cursor < 8)
      PositionLcd(8);
    else
      PositionLcd(0);
  }
```

(continued)

HCS II owners:
Here's that one little word you've been longing to hear...
ANNOUNCING THE HCS II VERSION 3

Now you can upgrade your HCS II to Version 3.0 and add these exciting features:

● New HCS/PC Interface
  • Query and override any system input, output, or parameter from the PC
➤ Revamped HOST Program
  • Use a mouse to resize, move, open, and close all HOST windows
  • Send messages from XPRESS to a HOST window for debugging
  • Set or clear system inputs, outputs, or X-10 modules and try out speech strings from HOST
➤ Modem Support
  • Call your HCS from a remote location and check its status, load a new XPRESS program, or retrieve logged data
● Caller ID
  • Access Caller ID data from XPRESS to announce or log who's calling
➤ Support for more digital I/O expansion boards
➤ Read and write eight netbits at a time with Netbyte
➤ Send messages to network modules directly from XPRESS
● Detect loss of AC power from within XPRESS
➤ Plus lots more for just $60



Circuit Cellar, Inc., 4 Park St., Vernon, CT 06066
(203) 875-2751 • Fax: (203) 872-2204

Listing 1—*continued*

```
  else
    DataWr(c);
  return;


PositionLcd(unsigned char c)  /* Position LCD cursor */
{
  Cursor = c;

  if (c > 7)
    CommandWr((c 8) + 64 + 0x80)
  else
    CommandWr(c + 0x80);
    delay(5);
  return;
}

ClearLcd(void)                /* Clear LCD and home cursor */
{
  Cursor = 0:
  CommandWr(1);
  delay(5);
  return:
}

DataWr(unsigned char c)       /* Write to LCD data register */
{                             /* and handle cursor positioning */
  if (Cursor == 8)
    PositionLcd(8);
  else if (Cursor == 16)
    PositionLcd(0);
  Cursor++:
  setbit(DRS)
  setbit(DEN)
  P1 &= 0x0f;
  P1 |= (c & 0xf0);
  clrbit(DEN)

  setbit(DEN)
  P1 &= 0x0f;
  P1 |= (c << 4);
  clrbit(DEN)
  return:
}

CommandWr(unsigned char c)    /* Write to LCD command register */
{
  clrbit(DRS)
  setbit(DEN)
  P1 &= 0x0f:
  P1 |= (c & 0xf0)
  clrbit(DEN)

  setbit(DEN)
  P1 &= 0x0f;
  P1 |= (c << 4);
  clrbit(DEN)
  delay(2);
  return:


InitLcd(void)                 /* Initialize LCD panel */

  clrbit(DRS)                 /* Put LCD in known state */
  delay(20);

  setbit(DEN)
```

(continued)

time I spent at the assembler level during the course of my custom library creation explains why I elect to code my LCD driver in C this time.

The first functions in the LCD driver module are conventional C implementations. PutStr displays a null-terminated string by merely passing characters off to PutChar until a null byte is encountered. PutChar outputs a character at a time and handles the new-line character by advancing the cursor to the beginning of the next line. All other characters are assumed displayable and are written to the LCD.

PositionLcd setsthecursor address to the value specified by the caller. Unfortunately, some translation is necessary to provide a number that is understood by the HD44780 LCD controller. Although it doesn't do it well, this universal LSI handles just about any small display configuration.

A result of this flexibility is that the logical-to-physical cursor correspondence is discontinuous with most LCDs. You get around the problem by tracking the logical cursor position and invoking a corrective maneuver when necessary. You can track the cursor by reading the LCD's status register or keeping a local copy for reference. Not wanting to waste a pin on the LCD's read/write line (it runs in write-only mode), I adopted the latter approach.

Here, the global register (in internal RAM) variable Cursor is used for this purpose. The resulting gyrations are performed before CommandWr is invoked to update the LCD's control register. An adjustment must be performed if the cursor address is greater then seven. In either case, the most-significant bit indicates to the LSI that this is a cursor set command. ClearLcd zeros the cursor address variable and invokes the LCD clear command via CommandWr.

DataWr performs the obligatory cursor fixup prior to splitting the data byte into nybbles (remember the LCD operates using a 4-bit bus) and falling through to the bit I/O level. Using Micro-C's extended preprocessor lets you use bit-manipulation macros that expand directly to 8051 SETB and CLR instructions. Here, clearing DRS selects the LCD's data register. DEN is toggled to generate the data strobe. CommandWr operates similarly. It has no cursor entanglements and selects the command register for its transfer by setting DRS high prior to clocking the nybbles across the interface.

The initialization function InitLcd begins at the nybble-oriented level since no assumption can be made of the operational status of the LCD at this time. The first three sequences ensure that the transfer mode is set to operate over 4 bits. This repeats three times so the command is recognized regardless of the operational mode of the LSI. Following this, the actual operating parameters are strobed into the controller using the standard CommandWr function.

DIGITAL TEMPERATURE

Temperature acquisition for the system is handled using the somewhat overqualified DS 1620 thermometer and thermostat IC from Dallas Semiconductor (also see Jeff's article in ***INK 42).*** Although it possesses more capability than is needed to measure temperature, it has the virtue of providing a purely digital interface to the host processor.

The DS1620 contains all temperature measurement and signal conditioning circuitry on-chip. It presents the processor with a three-wire digital interface composed of a bidirectional data line (DQ), reset (\RST), and clock (CLK). The temperature reading is in a 9-bit, two's complement format. The measurement range spans from -55°C to +125°C in 0.5°C increments. Table 1 shows the DS1620's output for several measured temperatures.

Data transfers into and out of the DS1620 are initiated by driving \RST high. Once the DS1620 is taken out of reset, a series of clock pulses is emitted by the processor to transfer the data. For transmission to the DS 1620, data must be valid during the rising edge of the clock pulse. Data bits sent to the processor are output on the falling edge of the clock and remain valid through the rising edge. Taking the clock high results in DQ assuming a high-impedance state. Pulling \RST low forces DQ into a

Listing *1-continued*

```
  P1 &= 0x0f;
  P1 |= (0x30);
  clrbit(DEN)
  delay(5);

  setbit(DEN)
  P1 &= 0x0f;
  P1 |= (0x30);
  clrbit(DEN)
  delay(5);

  setbit(DEN)
  P1 &= 0x0f;
  P1 |= (0x30);
  clrbit(DEN)
  delay(5);

  setbit(DEN)                 /* Set 4-bit mode */
  P1 &= 0x0f;
  P1 |= (0x20);
  clrbit(DEN)
  delay(5);

  CommandWr(0x28);            /* 4 bit, 1 line, 4 x 7 matrix */
  delay(5);

  CommandWr(0x0c);            /* Display on, cursor off */
  delay(5);

  CommandWr(0x06);            /* Auto increment, shift right */
  delay(5);

  ClearLcd();                 /* Clear the LCD and set initial */
  return:                     /* cursor address */
}
```

Listing 2—Primitive support routines for the DS1620 are written directly in assembly language.

```
*I/O bits
RST           EQU    P3.5
CLK           EQU    P3.3
DQ            EQU    P3.2

*Configure for CPU control, continuous conversion
TempConfigure
              SETB   RST
              MOV    A,#$C
              ACALL  TWR
              MOV    A,#%00000010
              ACALL  TWR
              CLR    RST
              MOV    R2,#20
              ACALL  DELAY
              RET

*Start temperature conversion
TempConvert
              SETB   RST
              MOV    A,#$EE
              ACALL  TWR
              CLR    RST
              RET

*Read temperature conversion
TempRead
              SETB   RST
```

(continued)

Listing **2**—*continued*

```
                MOV     A,#$AA
                ACALL   TWR
                ACALL   TRD
                CLR     RST
                RET

*local:  write 8 bits to DS1620
TWR:
                MOV     R0,#8
TWR1:
                CLR     CLK
                RRC     A
                MOV     DQ,C
                SETB    CLK
                DJNZ    R0,TWR1
                RET

*local:  read 9 bits from DS1620
TRD:
                SETB    DQ
                MOV     R0,#8
TRD1:
                CLR     CLK
                MOV     C,DQ
                RRC     A
                SETB    CLK
                DJNZ    R0,TRD1
                CLR     CLK
                MOV     C,DQ
                MOV     B.0,C
                SETB    CLK
                ANL     B,#$FE
                RET
```

high-impedance state and immediately terminates communications.

Temperature data is transmitted over the 3-wire bus in least-significant-bit first format. A total of nine bits are transmitted where the most-significant bit is the sign. If all nine bits are not of interest, the transfer can be terminated at any time by asserting \RST. Waveforms illustrating read and write sequences are shown in Figure 2.

The DS1620 support routines are in assembler (see Listing 2). The DS 1620 has a nonvolatile EEPROM configuration register (Table 2) of thermostatic and operational information. Since I'm not using thermostatic functions, I only need the operational parameters. TempConfig is hard coded to set the configuration register for operation under CPU control and continuous temperature conversion.

Once in continuous conversion mode, the actual conversion process is started by issuing the start conversion command through TempConvert. Using TempRead, the DS1620 can be read at any time and the last temperature conversion is returned. The result is returned in the 16-bit accumulator as defined by Micro-C consisting of the B (MSB) and ACC (LSB) registers.

MAINLINE GLUE

Invoking the support drivers is managed by the main module shown in Listing 3. This module takes control after the Micro-C startup routine finishes. On entry, the code instructs the DS1620 to start performing temperature conversions, initializes the LCD, displays the log-on message, and enters into an endless loop.

This loop continuously reads the DS1620, performs a Celsius-to-Fahrenheit conversion, translates the resulting binary number to ASCII, and displays the conversion result on the LCD. The temperature conversion is performed using the familiar equation:

$$F = C \times \frac{9}{5} + 32$$

Since the DS1620 returns temperature in 0.5°C increments, the value is first divided by two. Unlike the often impenetrable gyrations that

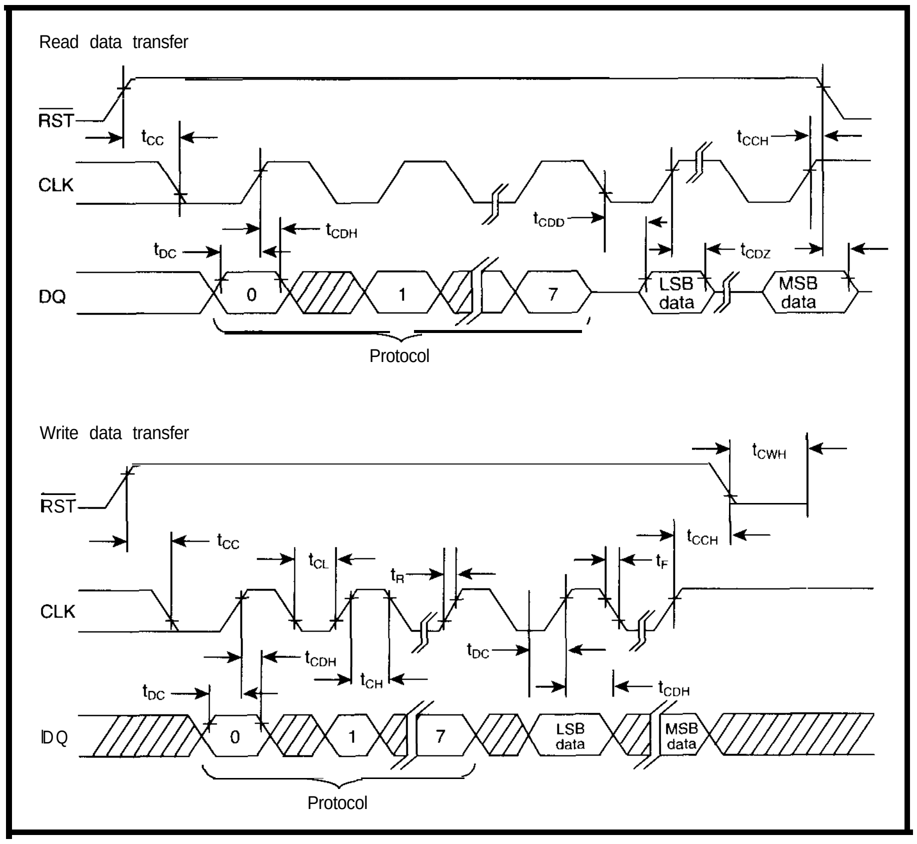


Figure **2**—*The DS1620's* read *and write timing waveforms. All communications are initiated by first releasing the reset pin.*

Temp. (°C)	Output (binary)	(hex)
+125	011111010	0FAh
+25	000110010	032h
+½	000000001	001h
0	000000000	000h
–½	111111111	1FFh
-25	111001110	1CEh
-55	110010010	192h

Table 1--The *DS1620 sends out a two's complement digital pattern that can be directly clocked into a microcontroller.*

result when working with numbers in assembler, the C rendition of this calculation is perfectly clear in intent and function. A short sequence of divisions, modulos, and logical OR operations results in decimal ASCII values that are dispatched to the LCD.

NOW I GET IT

I had only sparse specification for my investigation of the AT89C2051. Having worked with other small 805 1 derivatives, I was concerned about what deviations existed between the AT89C205 1 and a "full" 805 **1.** That is, I was looking for a run down of unsupported instructions and, more importantly, a list of instructions that would make the processor go haywire if inadvertently executed.

The official response was that the AT89C205 1 contained a standard 805 1 processing core and was capable of performing all 805 1 instructions (although some would obviously do nothing useful).

Having accidentally instructed other 8051 derivatives to execute instructions not in their repertoire, I'm aware of the consequences of such careless programming. However, the AT89C2051 appears to operate as advertised. The thermometer program on the protoboard is laced with long jumps and calls that execute nicely.

This brings up an important point. Since the AT89C2051 has only 2 KB of program memory, the entire address space could be navigated using absolute calls and jumps, which saves a considerable amount of memory.

Currently, I'm compiling under the tiny memory model, so the code generator automatically includes the proper startup code and run-time library and accesses the appropriate library files for any referenced functions. Although the tiny model assumes a single-chip environment and manages the meager memory resources appropriately, this is not enough to tell Micro-C whether it should perform branching using long or absolute addressing. In addition, to provide the necessary flexibility, the library functions are all written using long jumps and calls.

Micro-C applies an interesting approach to the code-generation process that proves to be remarkably effective. Assembler code is generated as the output of compiling (and, of course, is the source format of assembly language programs). The source linker combines these assembler modules, picks up the appropriate startup code and run-time library, and attempts to resolve any undefined references by searching the applicable library files.

Once the necessary files and various memory segments have been combined, compiler-generated labels are adjusted to be unique within each (input) file. The resulting file is a large assembler program. The process is completed by passing this assembler file through the absolute assembler to generate the final Intel or Motorola hex load image.

To some, this may look like it's backwards since the assembly process

Listing 3-*The mainline code for the* **simple digital** *thermometer using the AT89C2051 and DS1620 lets the support code do the real work.*

```
/* 8051 definitions  */
#include <8051reg.h>
#include <8051bit.h>

/* I/O bits */
#define Led P3.7

main(void)

  unsigned char c;
  unsigned int i;
  TempConvert();
  InitLcd();
  PositionLcd(0);
  PutStr("Mid-Tech");

  i = 0;
  while (1){
    PositionLcd(12);
    c = TempRead();
    c = (((c/2) * 9/5) + 32);
    if (c > 100){
      PutChar((c / 100)  | '0');
      c %= 100;

    else
      PutChar(' ');

    if (c > 10){
      PutChar((c  /  10)|'0');
      c %= 10;

    else
      PutChar(' ');
    PutChar(c | '0');
    PutChar(0xdf);

    if (++i ==  100){
      i = 0;
      cplbit(Led)
```

	DS1620 mode	Data	
ꞆPU mode	(3	(LSB first)	Comments
Tx	RX	0Ch	CPU issues Write Config command
Tx	RX	00h	CPU sets DS1620 up for continuous conversion
Tx	RX	Olh	CPU issues Write TH command
Tx	RX	0050h	CPU sends data for TH limit of +40°C
Tx	Rx	02h	CPU issues Write TH command
Rx	Tx	0014h	CPU sends data for TH limit of +10°C
Tx	Rx	Alh	CPU issues Read TH command
Rx	Tx	0050h	DS1620 sends stored value of TH for CPU to verify
Tx	Rx	A2h	CPU issues Read TL command
Rx	Tx	0014h	DS1620 sends stored value of TH for CPU to verify
Tx	Rx	EEh	CPU issues Start Convert T command

Table 2—*The DS1620 contains a nonvolatile storage region that is used for setting operation parameters and passing status information. The high and low temperature limits are held in fhis area.*

is the last step performed. I guess this just goes to show you that there are different ways to do things. However, there are advantages to generating a large monolithic assembler file late in the code-generation process. If you're so inclined, you can effectively monkey with the code generator and modify processes you could never get at in a conventional compiler.

The relevant point is that it's relatively easy to perform simple text substitution, eliminating undesirable instruction formats, since you've got a full assembly file. Previously, this was the only way you could generate functional code for processors that didn't support LJMP and ACALL instructions. Current versions of Dunfield's ASM5 1 require only setting a switch to completely automate the translation process.

The AT89C205 1 is destined to be an important addition to the small system designer's arsenal. Now that I've gotten through my initial AT89C-205 1 experience, I'm ready to put it to work in a serious application. ■

John Dybowski is an engineer involved in the design and manufacture of embedded controllers and communications equipment with a special focus on portable and battery-operated instruments. He is also owner of Mid-Tech Computing Devices. John may be reached at (203) 684-2442 or at john.dybowski@circellar.com.

SOFTWARE

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

I R S

422 Very Useful
423 Moderately Useful
424 Not Useful

Home Control is as simple as 1, 2, 3...

Energy Management
☆
Security & Alarm
A
Coordinated Home Theater
A
Coordinated Lighting
A
Monitoring & Data Collection

Get all these capabilities and more with the Circuit Cellar HCS2-DX. Call, write, or fax us for a brochure. Available assembled or as a kit.

1.) The Circuit Cellar HCS2-DX board is the brains of an expandable, network-based, control system which incorporates direct and remote analog and digital I/O, real-time event triggering, and X-10 and infrared remote control. Control programs and event sequences are written on a PC in a unique user-friendly control language called XPRESS and stored on the HCS2-DX in nonvolatile memory.

2.) The Relay BUF-Term provides hardwire interface protection to the HCS2-DX. Its 16 inputs accommodate both contact-closure and voltage inputs within the range of ±30V. Its 8 relay outputs easily handle 3 amps AC or DC, to directly control solenoids, motors, lamps, alarm horns, or actuators.

3.) The PL-Link provides wireless X-10 power-line control to the HCS II system. The PL-Link is an intelligent interface that sends, receives, and automatically refreshes X-10 commands. It works with all available X-10 power-line modules.

Of course, there's a lot more! The HCS II system has phone and modem interfaces, infrared remote control, voice synthesizers, and much more. Whatever your application, there's a configuration to accommodate it.

GET STARTED TODAY WITH AN HCS2 "123-PAK"

The 123-PAK consists of an HCS2-DX board, Relay-BufTerm board, PL-Link board, TW-523 powerline interface, PS12-1 power supply, and Ver. 3.0 HCS XPress software.

Assembled $676 Kit $516

CIRCUIT CELLAR, INC.

4 Park Street, Vernon, CT 06066
(203) 875-2751 • Fax (203) 872-2204

Prices shown are plus shipping.

#136

CONNECTIME conducted by Ken Davidson

The Circuit Cellar BBS
300/1200/2400/9600/14.4k bps
24 hours/7 days a week
(203) **871-1988—Four** incoming lines
Internet E-mail: **sysop@circellar.com**

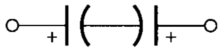
This month's topics are an interesting mix. In the first thread, we ponder the question of just how back-to-back polarized electrolytic capacitors really **work.** *As so often is the case when analyzing circuits, you need to look at an equivalent circuit to see what's really going on.*

Next, we try to **overcome** *the problems caused by those pesky AC power interruptions that are all too familiar during the summer months. Finally, designing video circuitry can be something of a black art We look at some alternatives for doing the genlock* **that is so** *necessary when designing a video titling box.*

Back-to-back caps

Msg#:38310
From: Tom Carter To: All Users

I don't understand the effect of connecting polarized capacitors in this back-to-back configuration.



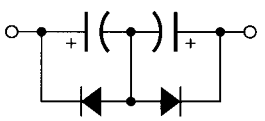
How does it work?

Msg#:44837
From: James Meyer To: Tom Carter

You may hear someone say that the caps work alternately as diodes on each half-cycle of the applied AC voltage. Don't believe this person.

For the following discussion, keep in mind that whenever you use caps like this, you should use two caps that are as near to identical as possible. Both should have the same capacitance rating and voltage rating. It's good if they can be from the same manufacturer and have the same date and lot code as well. The reasons for this will be made clear shortly.

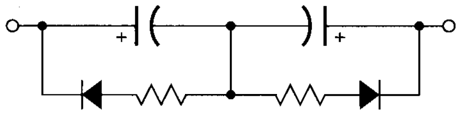
The result of paralleling each of the caps with a diode like this:



would be equivalent to a cap with a capacitance and voltage rating equal to one of the original caps.

What you will find if you actually connect the caps together like you originally showed, though, is that the resultant capacitance of the combination is . half . what either cap would be by itself. Clearly there is more going on here than simple diodes.

What a single electrolytic cap looks like is more like a perfect cap that has a diode and a resistor in parallel with it:



There are the equivalent of diodes in there, but they are diodes with fairly large-valued resistors in series with them. When the series capacitors have AC applied to them, the diodes gradually charge each cap up with DC. The DC voltage across each cap actually gets up to nearly half the peak-to-peak value of the AC voltage. That's close to 150 volts in a nominal 1 10-VAC RMS circuit.

If you look with a scope across either cap when the circuit has been in operation for a short while, you will see that each cap has AC and DC voltage on it. The DC voltage is large enough so that the sum of the AC and DC is such that there is (almost) never a time that the cap is reverse biased.

As to what happens to the current through a series-connected combination like this: All but a very tiny bit of the current flows through the caps. The current rating of back-to-back caps is not degraded by their connection. It is still equal to whatever current rating one of the original caps has.

You can probably see by now that it is important that the capacitors be matched in almost all of their electrical characteristics. If they were mismatched, they wouldn't charge up with DC equally and there would be a probability that one of the caps *would' get reverse biased and be damaged.

At one time, most of the unpolarized electrolytic caps available were simply two ordinary, back-to-back connected electrolytic caps in a case that was twice as long as an ordinary cap. I've taken enough of them apart.

As you know, the dielectric in a normal, DC, or polar aluminum electrolytic cap is a thin aluminum oxide film on *one* of the foil "plates." That's what's responsible for the different colors of the two foils that you see when you unroll a cap. In many modern bipolar aluminum caps, *both* foils have an oxide film so you get the equivalent of two caps in series even though there are only two foils and not four. The consequence of all this is that for the same capacitance and voltage rating, an unpolarized electrolytic cap will (must) be physically larger than its polar cousin.

If this isn't more than you ever wanted to know about connecting two caps in series, let me know. We've only scratched the surface. :-)

Short-glitch UPS

Msg#:18106
From: Tom Moran To: All Users

Here in Silicon Valley, CaliFEMA, our residential electric supplier, is getting ever less reliable. I've seen UPSs for computers that appear to run for tens of minutes for a couple hundred dollars, and I've seen generators that run on gas till the tank is dry. But most of our glitches are just a second-just long enough to reboot the computer, reset all the clocks, and so on. Are there devices that will keep a whole house running uninterrupted during those few-second flickers? For those times we do lose power for an hour or a day, are there computer UPSs that give you just a minute (to save files) and don't cost hundreds of dollars?

If a person wanted instead to go the whole way and have uninterrupted power, supplied by the electric company when available, or generated from gas otherwise, for the whole house, what kind of money are we talking?

Msg#:19340
From: Ed Nisley To: Tom Moran

The catch with a "whole house" UPS is that it must be able to support the maximum rating at your entry panel.. .say 200 A at 240 V. Finding a 48-kW UPS is, as you might imagine, a bit of a challenge. Worse is installing it, because it must switch over without a glitch. You don't have time to fire up a mechanical generator, which means the UPS must live "between" the service entry and your main breaker panel. Ugh.

You might consider running a separate set of "UPS outlets" throughout your house for things like clocks and PCs. Even though you'd use standard boxes with 15-A duplex outlets, you'd have only a few hundred watts maximum draw.. .which would work fine with a standard UPS. Plug a toaster into the thing and you get what you deserve.

I've done roughly that sort of thing here. There's a 1500-W UPS tucked into the crawl space with a line to my office and (pretty soon) another to Mary's desk, each terminating in a standard electrical outlet that's octopused with extension cords. The UPS is a ferroresonant thing that's noisy enough to deserve being banished to the basement; your mileage may vary. In any case, it supplies clean, full-time AC to the PCs and rides through summer-time power glitches without a burp.

We still must reset the microwave oven, but the bedroom clock has its own backup battery.

Msg#:19667
From: Ken Davidson To: Ed Nisley

Hospitals do just that. Most hospital rooms have red outlets on the walls. Those are uninterruptable outlets intended for life-support and other mission-critical equipment. Since theoretically nobody should die if a patient TV or bed lamp goes off in a power failure, those are plugged into the regular outlets.

Msg#:21878
From: Ed Nisley To: Ken Davidson

When we lived in Connecticut, I considered wiring the office with those isolated-ground "orange outlets" just to indicate that they went back to the UPS rather than the breaker panel. That fell flat when I discovered the Home Depot price was something like 25 bucks a pop; I'll just have to remember which is which.

How long has it been since I told the story about why IBM put their electrically heated ceramic kilns on the emergency backup power supply? They had a set of gas turbine generators in semitrailers dedicated to the cause.. .a wonder to behold!

Msg#:20768
From: Dave Tweed To: Tom Moran

The ultimate "short glitch UPS" is an MG (motor/generator) unit, such as IBM used to supply with their System/360 and System/370 mainframes. These consist of a motor spinning a flywheel which is in turn spinning a generator, from which power is supplied (all the time) to the load. When input power fails for short periods, the flywheel has enough energy to keep things spinning with essentially no drop in voltage or frequency. When power comes back, there's no phase discontinuity or sudden voltage change. (Actually, the motor armature, flywheel, and generator armature are all the same spinning mass, with care taken that there's no direct transformer coupling from the motor windings to the generator windings.) It's really the ultimate in clean, sine-wave power.

CONNECTIME

Now, the problem is that I've never seen these units available surplus. However, if you ever find one, that's what you want. Maybe you can build something similar from a motor and a large alternator. Look for a local electric motor repair service, and stop by to see what they've got in their (large) "junk box." You can get some real bargains that way.

For the ultimate short-and-long-term UPS, couple a gas motor to the shaft with a freewheel clutch, and fire it up as soon as the primary power fails. As soon as the motor speed catches up to the flywheel speed, you're on backup power, again with no glitches. Some really sophisticated systems even have a clutch that lets the flywheel (or a second one] spin up the gas motor instead of using a battery and starter motor.

Msg#:20802
From: Russ Reiss To: Dave Tweed

Well, whether or not a MG set is "the ultimate" depends a lot on whether you like motors running all the time. But I might add that for most PC applications, you would not have to worry a great deal about the precise frequency. In other words, it would be fine if the MG slowed down a bit under load (this might permit a lower rating). Nearly everything in a PC system is powered from DC derived from switching power supplies that typically can handle 47 Hz, rather than require a precise 60 Hz rate. For a few seconds, that wouldn't bother things much. Even a small drop in voltage from the 117 VAC nominal wouldn't bother it much. Just a thought.

Msg#:31488
From: Pellervo Kaskinen To: Dave Tweed

I tend to differ. To me the ultimate is simply added capacitors on the critical lines, at least on the +5- and +12-V supplies. For the 5 V, I would consider those 5.5-V, 1-F supercaps. Maybe they would not handle the ripple or something else, but at least I would consider.

I have an extended capacitor on my alarm clock. That came to be because I was annoyed by the frequent resetting required in my previous home. In the original location, there never were any glitches, but the next location was almost as bad as my present one. So I added the extra capacitor there. Now, the clock can be moved from one outlet to another and it resumes the correct time. Of course, the LEDs were blank during the move.

Nowadays, I never need to reset the alarm clock after the glitches that play havoc on the stereo clock and station memory. A long power outage, something like three times during 2½ years, is another story.

One of my earliest industrial jobs was to chart the loads and interconnections of two diesel back-up power systems in the first Finnish oil refinery. The back-up power was supplied to the mission-critical instrumentation. There was a 25-kW unit in the main power distribution center. Another 85-kW unit was located in the main process control center.

Both units had a heavy industrial diesel engine and a flywheel plus a clutch. The 85-kW unit was also equipped with a 15-kW motor to keep it spinning. In theory, either unit could have been running by the generator acting as a motor. The intention was that the clutch would engage any time the incoming power was lost. The flywheel would kick start the diesel engine and the generator would start acting according to its name.

Whatever the reasons, they did not want to run the systems in the way where the generator was on the net all the time. Therefore, the 15-kW motor kept the flywheel up to speed.

As part of my checking the systems, I went through testing the actual transfer operations. For the 25-kW unit, we had a manual crank. Took about 15 to 20 minutes of sweating to get the system up to speed, operating alone. With two guys it was considerably quicker, I think down to about 2 minutes. But closing the clutch was an impressive thing! The clutch screeched and screeched and the motor started saying putt...putt...putt. ..putt...putt...purrrr.

The engineer in charge of the instrument center was unhappy about a voltage-monitoring recorder indicating a prominent dip at the times of changeover. All of that was natural, because the generator, while idling, was not synchronized to the incoming power (that the recorder followed). Then, at the time of change over, it could be full 180 degrees out of phase, in addition to the natural time required by the loss of primary sensing devices. There was going to be a glitch, no matter what. All we could really suggest, was to filter the recorder a little more.. .

Msg#:22363
From: Lee Stoller To: Tom Moran

Well, I guess I can't resist mentioning what we have at NBC.. three large inverters support our equipment room, Switching Central and Skypath (satellite feeds) operations, plus other selected facilities. The inverters are connected to a bank of deep-cycle batteries that keep everything going long enough so that the twin diesel generators upstairs can get started and up to speed. It also isolates us from power glitches.

Certain things (like our Cesium standards) are fed from small (1250-watt) Clary UPS units, so they keep going even if the 'big* UPS system fails. (We wear belts *and* suspenders!)

CONNECTIME

Video genlock circuitry

Msg#:28562
From: Jim Gladney To: All Users

Has anyone had success designing a video genlock circuit with less than 10 nanoseconds of jitter? I've tried several PLL designs without too much luck, and have finally resorted to a brute-force divider circuit using a 104-MHz crystal. However, I fear it generates vast quantities of EMI.

Any suggestions would be welcome.

Msg#:30300
From: Lee Stoller To: Jim Gladney

Have you been able to identify where the jitter is coming from? Maybe it's in your measuring setup? Most normal genlock circuits will either work very poorly or very well.. .I don't generally see jitter that small in the video equipment that I work with.

Have you checked your power supply lines? Your jitter may be power supply noise getting into the PLL circuit. You may need more bypassing at the VCO (or other places). Most ordinary sync generators start at 14.318 MHz with a crystal oscillator and divide by four to make 3.58 MHz. As I said, phase noise such as you see is usually not a problem.

Msg#:31865
From: Jim Gladney To: Lee Stoller

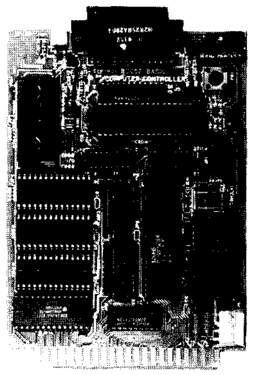
The jitter is due to my ineptness. You are right about PLL circuits either working very well or very poorly. My designs have been falling into the latter category.

The jitter is due to using an independent high-frequency oscillator and dividing down to obtain the pixel frequency. I would genlock by resetting the divider at the falling edge of the incoming horizontal sync. Primitive, but it worked for the application.

I would like to build an overlay titler that works directly on composite video. In order to manipulate the color phase, the circuit must synchronize with the color burst exactly. I realize that a PLL is the proper tool for the job, but have run into two problems:

1. If I use horizontal sync for the input frequency, the equalization pulses cause problems. Since these pulses are

BCC52
BASIC-52
Computer/Controller



The BCC52 controller continues to be Micromint's best selling single-board computer. Its cost-effective architecture needs only a power supply and terminal to become a complete development system or single-board solution in an end-use system. The BCC52 is programmable in BASIC-52, (a fast, full floating point interpreted BASIC), or assembly language.

The BCC52 contains five RAM/ROM sockets, an "intelligent" 2764/128 EPROM programmer, three 8-bit parallel ports, an auto-baud rate detect serial console port, a serial printer port, and much more.

PROCESSOR
- 80C52 8-bit CMOS processor w/BASIC-52
- Three 16-bit counter/timers
- Six interrupts
- Much more!

Input/Output
- Console RS232 - autobaud detect
- Line printer RS-232
- Three 8-bit parallel ports
- EXPANDABLE'
  - Compatible with 12 BCC expansion boards

Memory
- 48K RAM/ROM, expandable
- Five on-board memory sockets
- Either 8K or 16K EPROM

To Order Call 1-800-635-3355
Tel: (203) 871-6170
Fax: (203) 872-2204

BCC52	Controller board with BASIC-52 and 8K RAM	$1 89.00 Single Qty
BCC52C	Low-power CMOS version of the BCC52	$199.00
BCC52I	-40°C to +85°C industrial temperature version	$294.00
BCC52CX	Low-power CMOS, expanded BCC52 w/32K RAM	$259.00

CALL FOR OEM PRICING

MICROMINT, INC. 4 Park Street, Vernon, CT 06066
in Europe: (44) 0285-658122 • in Canada: (514) 336-9426 • in Australia (3) 467-7194
Distributor Inquiries Welcome!

#119

LOW COST VGA LCD CONTROLLERS

EarthVision/ISA — Earth/LCD-M

HIGH PERFORMANCE COLOR	***LOW COST MONOCHROME***
ISA 16 Bit VGA	ISA 8 Bit VGA PC
Windows Accelerator	Supports LCDs: 640x480
512K or 1M RAM	640x200, 320x200
CHIPS & TECH 65545	Hardware Panning
Fastest LCD Chip	Runs VGA Programs On
Controls All VGA LCD'S	On Low Res LCDs
Only $299	LOW cost - $149

FLAT PANEL KITS **(Includes controller, cable & backlight inverter -just plug it in!!)**

9.4" Monochrome	$249
8.2" Single Scan Color	$450
9.4" Dual **Scan Color**	**$749**
8.4" Active Matrix Color	**$849**
9.5" Active Matrix Color	**$995**

LCD's ONLY

9.4" Monochrome	**$95**
8.2" Single Scan Color	**$125**
9.4" Dual Scan Color	$399
8.4" Active Matrix Color	$595
9.4" Active Matrix Color	$750

EARTH Computer Technologies "The Flat Panel Solutions Company"

P.O. Box 7089 - Laguna Niguel - California - 92607
Phone: (714) 448-9368 - Fax: (714) 448-9316 - BBS: (714) 448-8093

#137

twice the frequency of the normal horizontal, they drive my PLL nuts.

2. If I use the vertical sync as my input frequency, the necessary divider is gigantic, something like 200k. The output of this circuit refuses to settle down.

I know this circuit has been designed a billion times. I suspect the input frequency used is either horizontal or the color burst itself (via the color burst gate of an LM1881).

If you could point me in the right direction, it would be greatly appreciated.

Msg#:32334
From: Lee Stoller To: Jim Gladney

The basic mistake (which you are already aware of) is using HSYNC to lock your high speed stuff. You must lock your 14.3 18 MHz (or whatever multiple you use) to the incoming color reference (3.58 MHz). You then divide down this to give your own HSYNC and VSYNC signals. In order to lock your picture horizontally and vertically, you need . two more* genlock circuits. These detect the incoming sync signals and goose your divider chains (issue reset pulses to your counters) to achieve this.

Your best bet may be to try to find a commercial NTSC sync generator on the surplus market somewhere.. .it can consume a lot of time if you are reinventing the wheel.

Msg#:32395
From: Jim Gladney To: Lee Stoller

Reinvent the wheel I must. I need to produce several hundred of these buggers.

Your statement of needing several genlock circuits did ring a bell. There is a sync regenerating IC from Harris called the CD22402. It can genlock onto an incoming video signal and "freewheel" when the sync is lost. Cable pirates love the thing.

I believe one of its outputs is a "pure" horizontal 15.57 kHz (no restoration pulses). I didn't use it because I didn't trust the timing accuracy of the 22402 genlock. But as you say, it must either work very well or very poorly, and it does work. I will go check the timings on the scope.

If this works, I would rather use horz than the burst gate, since I believe the color burst gate vanishes during vertical blanking.

If this does work, I still do not trust my PLL circuit. Have you any experience with the PLL.EXE 4046 design program available in the file area here? What I need is "4046 PLL design for boneheads."

Msg#:32797
From: Timothy Taylor To: Jim Gladney

You might want to take a look at the CA3126 from Harris (used to be an RCA chip). You feed it filtered video and a backporch pulse and it will give you a phase-locked continuos 3.58-MHz clock. You can then use this as your color phase reference or divide down for other apps. I've used this little chip many times before and it works well. The beast is cheap too. I think it's around $1 from Mouser. Use an LMI881 to get the backporch (a.k.a., burst gate) pulse.

Msg#:36341
From: Lee Stoller To: Jim Gladney

A word of warning: getting your 3.58 MHz from that chip is good, but dividing that 3.58 MHz for other purposes (I'm thinking about sync here) may or may not work, depending on what you want to do. It definitely will NOT give you standard pulse relationships. For correct NTSC-standard video, you must divide 14.3 18 MHz or your sync-subcarrier timing relationships will be wrong. There are *four* distinct fields in each color frame, and certain pieces of professional TV gear will kick if they can't identify them.

If closed-caption encoding/decoding is what you are doing, you must at least make pulses good enough to reliably allow you to tell the difference between even and odd fields. Your scheme seems to depend on the incoming genlock video to be standard NTSC, with your regenerated sync pulses "following along" to give correct timing. This dependence may be dangerous if the generated signal is to be used for broadcast purposes.

I don't want to seem too negative. What you are doing may make these concerns entirely irrelevant. Just thought I'd throw my 2 cents in.

Msg#:36728
From: Jim Gladney To: Lee Stoller

Lee, you are right, the last thing you want to do for closed captioning is to regenerate the sync. If you get out of phase with the four-field business, there indeed would be hell to pay. Not to mention wrecking whatever else is on the vertical blanking interval. I have followed a strategy of leaving the signal alone as much as possible then "punching in" the data at the right spot.

For this project, I'm building an "open" caption encoder. Essentially it just overlays text over video. I don't want to use an off-the-shelf text inserter IC since the fonts are too ugly.

Msg#:36731
From: Jim Gladney To: Timothy Taylor

Have you seen the new Motorola MC44144 chip? It is similar to the CA3 126 in that it is a subcarrier PLL, except this guy produces both 3.58- and 14.318-MHz outputs locked to a composite video input. I can use the 14.318 to create a pixel clock for my overlay project. It is an 8-pin DIP and requires only several external components (crystal, caps, resistors). I don't have pricing yet; I just happened to run across a data sheet/advertisement in the *EE Times*.

Msg#:43191

From: Timothy Taylor To: Jim Gladney

Sounds like an interesting chip. There's only one problem with these type of chips, though (at least with the CA3126): the video signal must be color NTSC. A mono or black-and-white show that does not have a color burst will produce a free-running clock output. It used to be that TV stations 'had' to suppress the color burst whenever airing that show. I believe the FCC now allows stations to air black-and-white shows with color burst, but there still might be some stations that are still suppressing it. The percentage of stations actually doing that is probably pretty low, though.

We invite you to call the Circuit Cellar BBS and exchange messages and files with other Circuit Cellar readers. It is available 24 hours a day and may be reached at (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, no parity, and 300, 1200, 2400, 9600, or 14.4k bps. For information on obtaining article software through the Internet, send E-mail to info@circellar.com.

ARTICLE SOFTWARE

Software for the articles in this and past issues of *Circuit Cellar INK* may he downloaded from the Circuit Cellar BBS free of charge. For those unable to download files, the software is also available on one 360 KB IBM PC-format disk for only $12.

To order Software on Disk, send check or money order to: Circuit Cellar INK, Software On Disk, P.O. Box 772, Vernon, CT 06066, or use your Visa or Mastercard and call (203) 8752199. Be sure to specify the issue number of each disk you order. Please add $3 for shipping outside the U.S.

IRS

425 Very Useful 426 Moderately Useful 427 Not Useful

DESIGN CIARCIA WORKS

Does your Big-Company marketing department come up with more ideas than the engineering department can cope with? Are you a small company that can't afford a full-time engineering staff for once-in-a-while designs?

Steve Ciarcia and the Ciarcia Design Works staff may have the solution. We have a team of accomplished programmers and engineers ready to design products or solve tricky engineering problems.

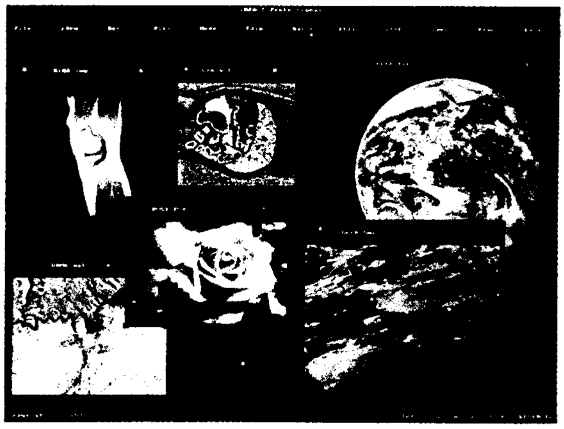
Whether you need an on-line solution for a unique problem, a product for a startup venture, or just experienced consulting, the Ciarcia Design Works is ready to work with you. Just fax me your problem and we'll be in touch.

REMEMBER...A

CIARCIA DESIGN WORKS

FAX: (203) 871-8986

A Serious Imaging Solution



IMPACT Professional is a complete image analysis system that includes a broad range of cross-discipline tools grouped into eight separate processing environments These modules include

- Image Processing System
- 2-D Analysis System for Edge. Shape and Surface Characterization
- Object Counting. Analysis and Sorting System
- Neural Network Graphic Recognition and Classification System
- Image Comparator and Anomaly Detection System
- 3-D Mapping System including Stereo- and Random-dot Projections
- Fractal Analysis System for Point. Line. Edge and Surface Processing
- Color Mapping System which handles Greyscale. Pseudo and 24-bit RGB

SYSTEM REQUIREMENTS: PC/AT or compatible. 386, 486 or Pentium, with at least 16 MB of RAM and a hard disk DOS 3 1 or higher Uses a flat memory model with its own extender and Virtual Memory Manager capable of addressing 4 gigabytes of memory. A super VGA video card.

TARDIS Systems
PO. Box 1251
Los Alamos. NM 87544 U.S.A.

FREE DEMO

Phone: (505) 662-9401
Fax: (505) 662-6780
Technical Support: (505) 662-5623

#138

STEVE'S OWN INK

Pyramid Schemes

In our everyday world where high tech is the nucleus, it is hard to think of life without the pervading influence of technology. We move so fast trying to accomplish whatever it is we do that we tend to evaluate the performance and aptitude of others on the same basis as ourselves. Reality is often something quite different. Those of us who live the fast, high-tech life often forget that it's just one big pyramid. Worse yet, it's an inverted pyramid, dependent on a constant source of a few basic elements.

Like most of you, I recognized early that the critical balance point in all this high tech is electricity. Having an independent source is mandatory, so I installed a big diesel generator and blissfully joined the ranks of the fat, dumb, and happy.

Last week a rude awakening toppled my pyramid. No water!

The good news about suburbia is the lack of congestion and the room for expansive living styles. The bad news is that your connection to the rest of society is through a couple copper wires and a narrow causeway. How many alternate support systems you build depends on your vulnerability to interruptions of these connections. I thought I had covered them all-all except water!

Halfway through the morning shower, the water stopped. Suspecting the well pump, I called a pump service and left for the office. After a few hours, I checked on their progress. They had extracted the apparently lightning-damaged pump, inserted a new pump and hose, got it stuck at the 200' level, ripped the hose off, and perhaps caved in part of the well. They suggested that now, instead of pump service, I needed a well driller!

I had never actually seen a well drilled before, so after calling the well guys, I decided to monitor their activity. Fully prepped from the movies, I had visions of high-tech drilling platforms, diamond-tipped bits, and hydraulic wonders. These fantasies instantly vanished as the ancient, rusting, well-drilling rig rumbled up the driveway. Its appearance suggested that the invention of the diesel engine and mobile drilling platform occurred at the same time in the late 19th century.

Thoughts of diamond bits and hydraulic drilling were dashed as they positioned a 2-ton, 20' x 5" pointed pipe-like-thing over the well pipe opening and then released it to drop as fast as it would go. As it hit bottom or whatever was in its way, there was a loud thud and slight tremor reminiscent of a low-Richter earthquake.

Apparently, there are various ways to drill wells-what you read about and what they actually do. This tried-and-true technique was called pounding a well and, in this case, a stuck pump.

After a day and a half of incessantly hammering up and down the well shaft, they abruptly stopped and simply said, "Ya' got watta' now. Gotta bail the well."

Bailing means pumping out a well to see how fast the water refills it and the pipe. I visualized some sophisticated, high-pressure pump and ultrasonic level-monitoring device. Instead, they dropped something resembling a 20' empty can down the well and repeatedly pulled it up and dumped the water.

I nearly came unglued over their level "sensor." I watched in horror as the well driller took a brick tied with a dirt encrusted hemp rope and dropped it down the well. The change in level was determined by jiggling the rope up and down, so you could hear the splash and measure the difference in rope lengths with a yard stick. "Got 'bout ten gallons here," he said as he pulled up the dirty rope. Apparently, that was good.

A short time and a few gallons of chlorine later, a new pump began providing water again. The experience, however, left me pondering the contrasts between my life and that of the well driller.

Those of us in computers struggle daily to keep up with a technology. It helps us better control our lives and be less vulnerable. In reality, however, we are still intensely dependent on a technology that is barely out of the stone age. The only aspect of it that seems to have kept up with the times is the cost. At $3500 for a little water, I wonder if I'm in the wrong business.

Steve