Fast emulator debugs 8085-based microcomputers in real time

New module even checks contents of intelligent peripheral chips and monitors activity at other critical circuit locations

by Michael Yaotung Yen,
Intel Corp., Santa Clara, Calif.

□ Spurred on by the frantic pace of semiconductor technology, microprocessors are faster than ever, and peripheral chips are so smart that some of them are more complex even than the processors they serve. To keep up with these advances, the Intellec Microcomputer Development System has been updated with the ICE-85 in-circuit emulator.

In-circuit emulation in effect extends the debugging capabilities of a software development system to the prototype microcomputer system (see "The ICE module reviewed," p. 110). By enabling the hardware and the software to be developed concurrently, the first generation of such development systems improved significantly on earlier methods of prototyping microcomputers. But the high speed of new microprocessors like the 8085 and the complexity of prototyping with smart peripheral chips required redesign of the plug-in ICE module, and the first of this second generation (pictured below) now allows much more rapid development of systems based on the 8085 microprocessor.

A series of hardware and software innovations provides the second-generation ICE-85 module with the following capabilities:

- Real-time emulation of the operation of the high-speed 8085 processor.
- User-tailored in-system diagnostics for use on peripheral chips.
- Significantly increased logic-analysis capability compared with previous ICE modules.
- Enhanced symbolic debugging.
- Display of the contents of the trace buffer in assembler mnemonics.

The problem of speed

One of the most important features of an ICE is real-time emulation of the processor, which allows the prototype system to operate at its full speed when using ICE. As the speed of microprocessors increases, this feature becomes difficult to achieve, primarily because of delays

in the signal cable. Previous emulators, typified by the ICE-80 developed for the 8080A, can operate only up to about a 2-megahertz clock rate, whereas the ICE-85 emulation module can run at the 8085's 3-MHz rate or even at higher speeds.

The first-generation ICE is shown in Fig. 1. The ICE cable plugs into the microprocessor socket in the user's prototype system, and there is about 6 feet of cable between it and the emulating microprocessor chip. Because of the length of this cable, the associated signal propagation delays to and from the emulating processor must be compensated for by a faster central processing unit in the emulated hardware. The ICE-80 module uses this technique to accomplish real-time emulation. However, in the case of high-speed processors such as the 3-MHz 8085, the propagation delays in the connecting cable are so significant that they can no longer be compensated for simply by installing a faster processing unit.

Close to the action

The ICE-85 module solves this problem by locating the emulating microprocessor chip on the plug at the end of the ICE cable, rather than inside the development system (Fig. 2), and it also has a significant portion of the emulator circuitry in a nearby cable box built into the cable assembly (Fig. 3). Thus, time-critical signals to and from the prototype memory and peripheral circuits reach the microprocessor chip without traveling through any cable. Only those signals requiring data multiplexing will incur a 1-foot cable delay to the cable box. However, since no real-time emulation signals travel on the 5-foot cable and buffer, these critical delays are essentially eliminated.

Mounting the microprocessor on the plug provides another important feature. The clock crystal circuitry of the 8085 processor chip is sensitive to an external capacitive load, and cable connections should be avoided. Since the processor chip is mounted on the cable plug, no cable connection is necessary to these crystal pins, and thus the ICE-85 module can operate from the crystal in the prototype system, which aids in checking out that circuitry. Also in the cable plug assembly is some simple multiplexing circuitry that directs data flows between the microprocessor chip, the prototype system, and the ICE cable box.

Peripheral chips are becoming highly integrated and also now require ICE-like diagnostic tools. Although an ICE could be developed for each peripheral chip, such a move is rather impractical. An ICE is a sophisticated product, and procurement of multiple development systems to house multiple ICE modules would be expensive. Also, synchronization of the multiple ICEs would be a difficult task, and controlling several modules from separate Intellec consoles would be confusing to a good many designers.

Another problem is the protocol that is used by peripheral chips to communicate with the central processor and which may require multiple input/output or memory accesses or both. With earlier ICE designs, a designer had no convenient way of executing the protocols for interrogating and changing the internal status of peripheral chips. The ICE-85 module, however, extends its diagnostic capabilities beyond the 8085 processor chip to include coverage of the peripheral chips, allowing total system diagnostics.

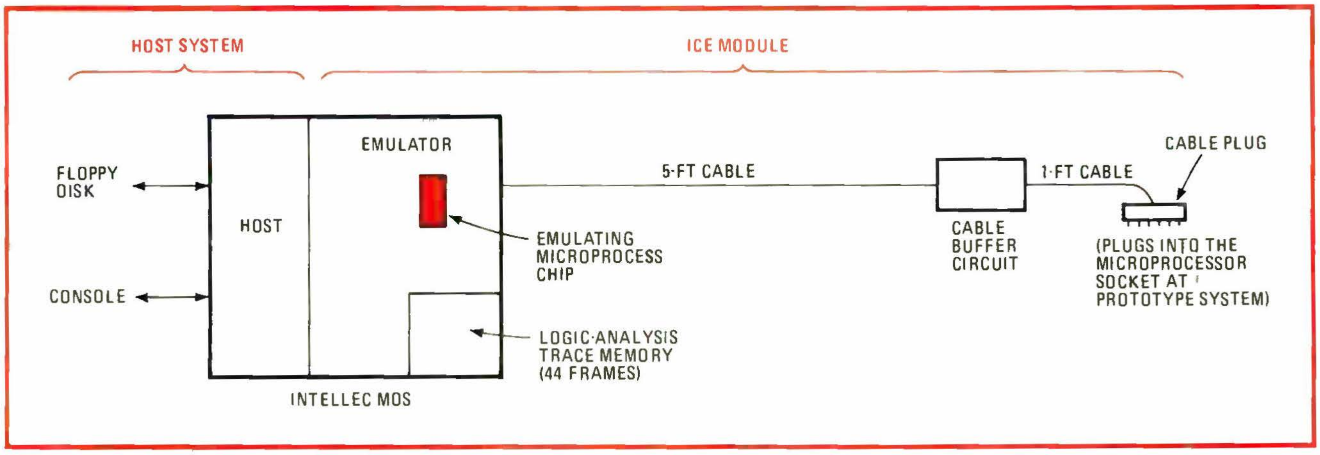
Handling the peripherals

First, ICE-85 software provides a "user macrocommand" capability to cover the processor routine required to change the peripheral chip's operation mode, change the chip's data-output ports, read the chip's status, or read the chip's data-input ports. The user defines this command routine and furthermore can assign it a symbolic macroname. Under this name, the macrocommand will be accepted at the ICE console and then executed in a manner transparent to the execution of the user development program.

This concept of symbolic control of peripheral chips can be further expanded. Since one macroroutine may be defined by a designer to access other macroroutines, it is enough to enter one simple high-level macrocommand at the console to display the contents of one or more peripheral chips.

A designer thus can type a simple macrocommand and cause ICE to interrogate and display the status of the

1. First generation. The ICE-80 module, for development of systems based on the 8080 microprocessor, has the emulating microprocessor inside the Intellec mainframe and linked to the prototype by 6 feet of cable. But cable delays limit emulation speed.


HOST SYSTEM
ICE MODULE
FLOPPY DISK
CONSOLE
HOST
EMULATOR
EMULATING MICROPROCESS CHIP
LOGIC-ANALYSIS TRACE MEMORY (44 FRAMES)
INTELLEC MDS
5-FT CABLE
CABLE BUFFER CIRCUIT
1-FT CABLE
CABLE PLUG
(PLUGS INTO THE MICROPROCESSOR SOCKET AT PROTOTYPE SYSTEM)

The ICE module reviewed

An in-circuit emulator creates a complete microcomputer development system by operating in conjunction with a host system generally used for software development. The ICE module connects to the user's prototype, extending the host system's capabilities to the hardware prototyping and software-hardware integration of the microcomputer. With ICE, the prototype can be debugged through the development system while it is trying out developmental programs that may reside either in the host system memory or in program memory on the prototype board.

The operation of the prototype's microprocessor is emulated by a similar microprocessor in the emulator. The module's support circuitry and software add diagnostic capabilities that would be impractical or impossible to add to the prototype itself. With an ICE module installed, the designer can run his developmental program in real time on the prototype.

An ICE generally operates in three modes: interrogation, run emulation, and single-step.

The interrogation mode occurs whenever user code is not being executed. In this mode, the user can investigate the results of the last operation and prepare the prototype system for a new operation by:

- Displaying or altering the internal registers, program counter, and stack pointer of the microprocessor and the memory locations or input/output ports of the prototype microcomputer system.
- Specifying an emulation breakpoint for a number of different conditions, such as memory read, memory write, instruction fetch, or I/O operation at a selected address, or any access to a user-defined nonexistent memory space, and examining the condition that causes a break in emulation at that point.
- Displaying the trace data that a high-speed trace memory has stored for a number of instruction cycles.
- Displaying the execution time measurement.

The run emulation mode executes the user's program and traces and compares information about each instruction. When a break condition is encountered or a time-out condition is met, emulation stops, and ICE returns to the interrogation mode.

The single-step mode runs the user's program and traces and compares information, but does so one cycle at a time. The emulated processor no longer runs at full speed; however, the emulator can gather much more detailed information about the program flow in this mode.

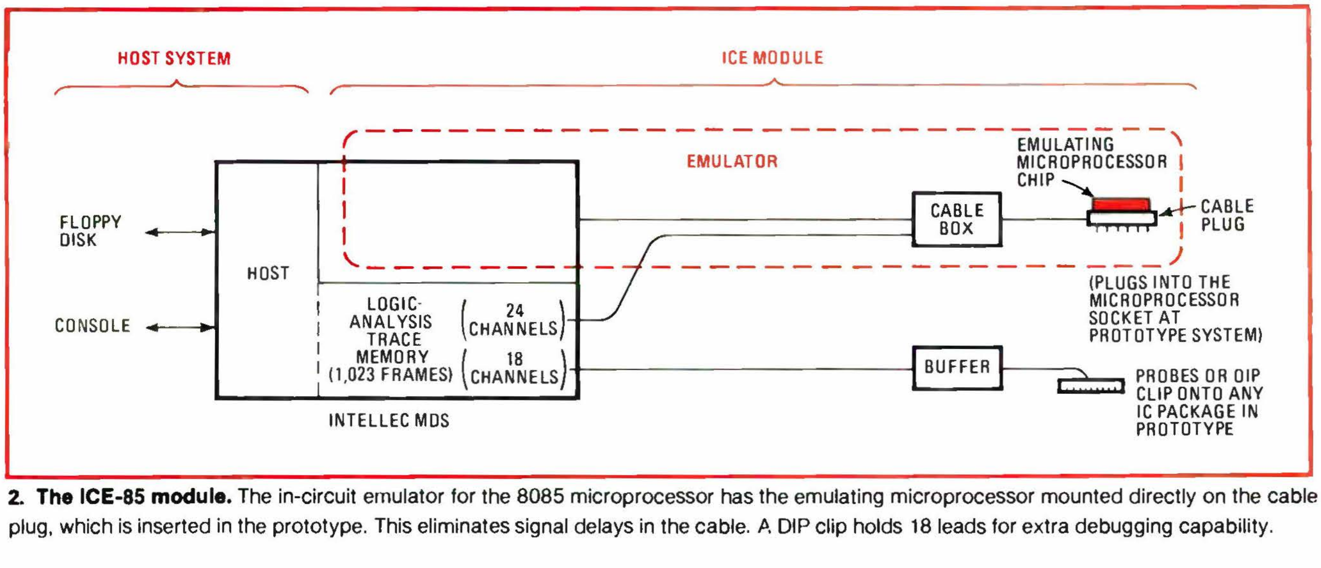
In-circuit emulation also allows the user to assign the development system resources to his prototype system. That is, he can use the random-access memory and I/O capabilities resident in the development system as though they were local memory and I/O in his prototype system. This feature allows the user to run his program in his prototype system, even before his prototype memory is built. In addition, it saves development time spent in temporary programming of a programmable ROM.

The host development system uses ICE software to load the object code of the user's program from an external mass-storage medium, such as a floppy disk, into either the development system memory or the user's prototype system RAM. The code is executed, debugged, and then returned to the development system's mass-storage device for later debugging sessions, PROM programming, or generation of ROM bit-patterns.

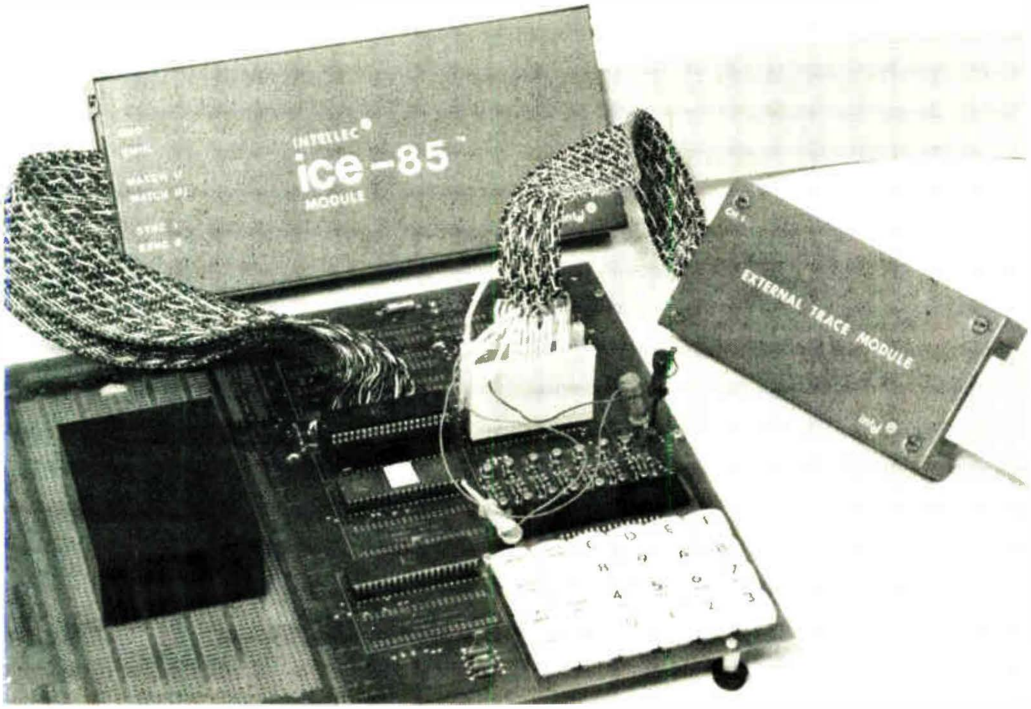
The memory emulation and program load/save capabilities allow the user to start running and debugging his prototype at the earliest stage of the hardware development cycle. As a result, design errors are quickly located and corrected, both hardware and software are debugged concurrently on the same prototype system, and the task of integrating hardware and software as a complete system is thus simplified.

The emulator also does extensive self-diagnostics. During a run emulation, it checks for the absence of clocks, and it verifies data written to user memory (by reading it back after writing). Errors detected will cause a disruption in the operation, followed by the output of the corresponding error message to the user.

Although ICE offers many valuable features, the physical connection from ICE to the user prototype system remains very simple—one cable plug to the microprocessor socket in the prototype system. No temporary jumpers, trace cuts, or circuit modifications are required to run the prototype system with ICE. This simple interface connection serves as a significant attraction to the design engineer looking for a highly efficient as well as convenient developmental debugging tool.


HOST SYSTEM
ICE MODULE
EMULATOR
EMULATING MICROPROCESSOR CHIP
CABLE BOX
CABLE PLUG
FLOPPY DISK
HOST
CONSOLE
LOGIC-ANALYSIS TRACE MEMORY (1,023 FRAMES)
24 CHANNELS
18 CHANNELS
INTELLEC MDS
BUFFER
(PLUGS INTO THE MICROPROCESSOR SOCKET AT PROTOTYPE SYSTEM)
PROBES OR DIP CLIP ONTO ANY IC PACKAGE IN PROTOTYPE

2. The ICE-85 module. The in-circuit emulator for the 8085 microprocessor has the emulating microprocessor mounted directly on the cable plug, which is inserted in the prototype. This eliminates signal delays in the cable. A DIP clip holds 18 leads for extra debugging capability.



3. Plug in. The ICE-85 module has its emulating microprocessor mounted on the cable plug, and 18 trace probes can pick up other signals directly from the prototype circuitry.

entire prototype system, giving information on the:

- Current instruction address.
- Current instruction.
- Data transferred during the instruction cycle.
- Current operation mode of all peripheral chips.
- Current contents of internal registers on all peripheral chips.

Further, when ICE is operated in a single-step mode, the designer can see how the peripheral chips change their internal status at the execution of each instruction. In this manner, he can quickly verify the peripheral control routines in his program.

In addition to its on-line trace memory for logic-state analysis of the 8085, ICE-85 also has an external trace module with 18 probes. The data captured by the probes synchronously with the clock can be used to break emulation under conditions specified by the user. These probes may either be used individually or be connected to a dual in-line package clip, which can be moved around the system to clip onto any integrated-circuit package. This feature allows ICE to monitor those signals that are of interest but not accessible through the user program alone.

As an example of the ICE-85 module's peripheral control, consider the 8253 programmable-interval-timer chip, one of the peripheral chips that may be used with the 8085 microprocessor. It consists of three independent, general-purpose, multimode 16-bit timers. Writing a control word to the chip selects one of the three timers in the chip and selects one of the six possible modes in which the timer may operate. After a control word is written to select an operation mode, a consecutive two-byte write operation specifies the initial count of the 16-bit timer. In such cases, the designer will want to know if the initial count is correctly set and will want to verify and reference the timer operation during his program's execution.

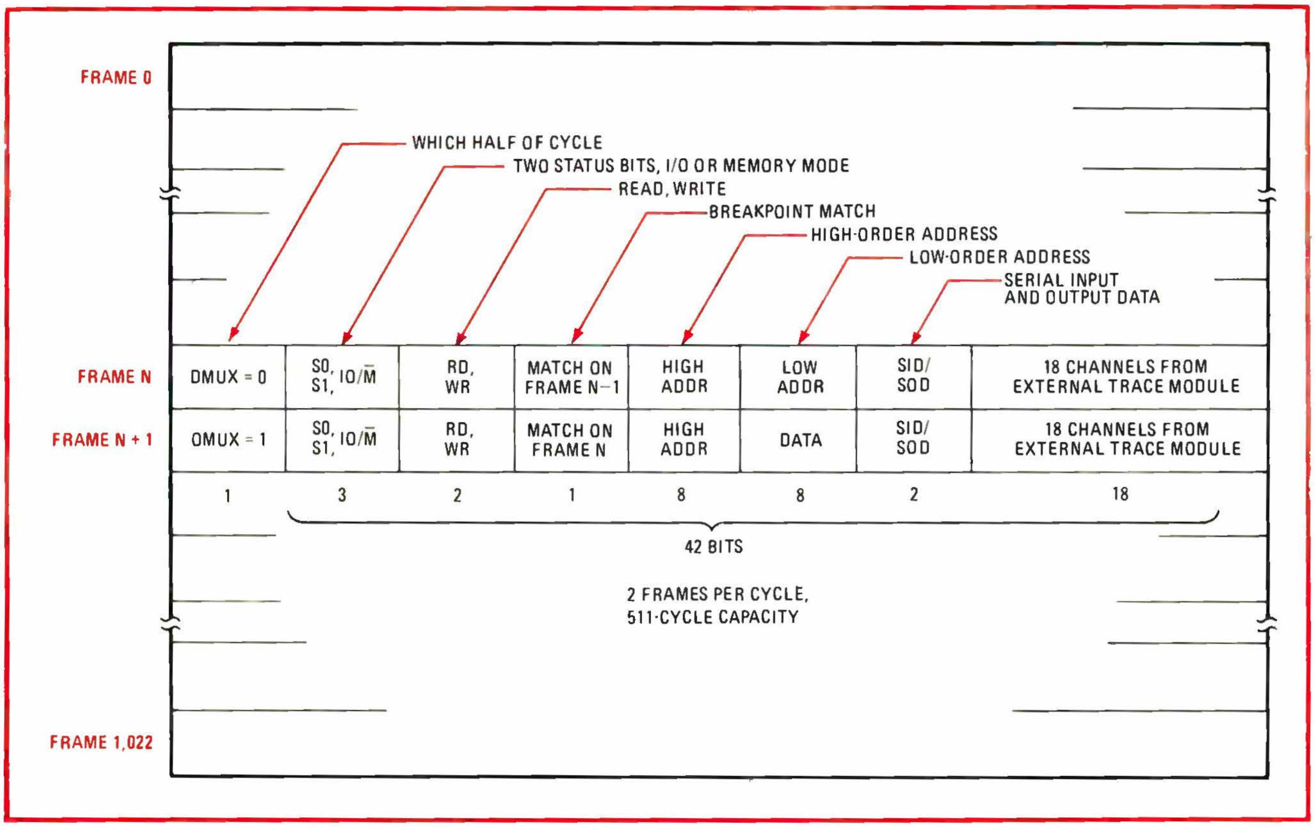
With the ICE-85 software, the design engineer can define macroroutines to control the timer. For example, he may symbolically initialize the timer, with a simple macrocommand, such as ":WTIMERO 32168". ICE then executes the macroroutine called WTIMERO, which initializes the 16-bit content of timer zero in the peripheral chip to be 32,168. He then directs the ICE to execute a segment of the development program until it encounters a breakpoint. At that point he may interrogate the current contents of timer zero by the simple command: "RTIMERO," which reads and displays the current content of the timer.

It should be noted that since the execution of peripheral routines is transparent to the execution of the user developmental program, the user program can resume execution after a break as if these routines were not inserted. Also note that control of the peripheral activity includes the transistor-transistor-logic circuits surrounding the processor in addition to the complex peripheral chips. Thus, the designer can obtain a total picture of his system's operation.

Logic analysis

The ICE-85 module combines in-circuit emulation with an enhanced logic state analysis. The earlier ICE-80 module has an on-line trace memory of 44 32-bit words. Figure 4 shows ICE-85's 1,024-word on-line trace memory, where each word, or frame, is 42 bits, or channels, wide.

Of the 42 channels of memory, 24 channels are assigned to the 8085 processor signals (such as 8 bits for



FRAME N	OMUX = 0	S0, S1, IO/$\overline{M}$	RD, WR	MATCH ON FRAME N−1	HIGH ADDR	LOW ADDR	SID/SOD	18 CHANNELS FROM EXTERNAL TRACE MODULE
FRAME N + 1	OMUX = 1	S0, S1, IO/$\overline{M}$	RD, WR	MATCH ON FRAME N	HIGH ADDR	DATA	SID/SOD	18 CHANNELS FROM EXTERNAL TRACE MODULE
	1	3	2	1	8	8	2	18

4. Trace memory. For logic analysis, the ICE-85 trace memory holds 1,023 frames of 42 bits each. Since the 8085 works on the basis of two frames per cycle, with low-order addresses and data multiplexed in alternate frames, the memory can store up to 511 machine cycles.

high-order addresses, 8 bits for either data or low-order addresses, and 1 bit for each of various functions such as status, read, and write and for serial data input and output) while the remaining 18 channels sample data from the external trace probes. The user can specify the grouping and formatting of trace data displays. This feature gives an in-depth and total picture of how an 8085-based prototype system operates.

In addition, there are two 42-bit breakpoint registers, plus two 42-bit clock qualifier registers that set up conditions for ending and advancing a trace operation. In the breakpoint and qualifier registers, each bit can be set, from the console, as either 0, 1, or "don't care." Thus, with 42 bits, a wide variety of conditions from the 8085 program and any other chip in the prototype can be used to perform breakpointing and tracing.

Symbolic debugging and mnemonics

Symbolic debugging was introduced in the ICE-80 software package, and it is also incorporated in an enhanced form in ICE-85 software. It allows the user to make symbolic references to the instruction and data addresses in his program. As the following example indicates, this feature greatly facilitates the use of relocatable object modules of code by the user during the program development cycle. Users need not be concerned about address changes during each reassembly and corresponding linkage of code modules. They need only reference symbolic labels initially attached to instructions in programming source code.

On the basis of the processor's address and data signals captured in trace memory, ICE-85 converts the operation codes of the executed instructions back to assembler mnemonics for display. This feature allows the designer to verify the actual instructions, in comparison with the assembly listing of his program.

To illustrate these points, the symbolic labels in the ICE commands make it easy to reference the instruction addresses in a development program. Without this feature, the designer would have to keep continual track of absolute addresses of subroutines as they change during the design process when new instructions are inserted. For example, a single command statement "GO FROM .LABEL1 TILL .LABEL2 EXECUTED" will cause the ICE to do all the following tasks:

- Set the program counter of the microprocessor to the absolute address symbolically labeled as LABEL1.
- Set a hardware breakpoint register to be the absolute address symbolically denoted as LABEL2.
- Run the processor in real time.

When the instruction at LABEL2 is executed, ICE will stop the processor, display a breakpoint message to the user, and return to the interrogation mode.

To sum up, the ICE-85 module can quickly debug systems built around the 8085 microprocessor. In the world of increasing peripheral-chip complexity, it reaches into the user's entire prototype system, providing active control and interrogation of the internal registers of the peripheral chips. Finally, with the 18 external probes, it allows monitoring and breakpointing capabilities to be extended to the IC external pin signals observed in the user's prototype system. □