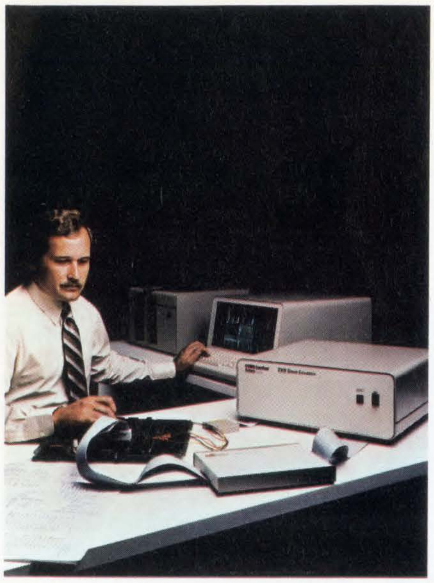


New-Generation μC Development System

Supports Development of Real-Time and Multi-Processor Systems

Lester G. Matheson and
David J. Ulmer
GenRad/Futuredata, Los Angeles

Today's μC development systems essentially are bundles systems. Since system hardware and software features are in one box, compromises were made.

The compromises exist due to economic considerations. At that time, μP components and memory were still quite expensive, and this had a strong effect on system design. This caused systems to appear as bundled, resource sharing, time sharing, bus sharing, memory sharing, processor sharing systems. With all this sharing going on, it was inevitable that the user and user's system had to share resources with the development system. In doing this, the compromises of the development system design were propogated into the user's system. The user's ability to develop modern multi-processor and real-time systems was therefore limited by the outdated architecture of the development system tools.

The development of the GenRad/Futuredata 2302 Slave Emulator is the first serious attempt to revolutionize the architecture of development systems and provide the user with vastly improved development capability. With these new tools a μC systems designer will be able to do a much better job of designing the systems of the future.

The slave emulator

The Slave Emulator, therefore, was designed to unbundle this outdated architecture. The goals were to develop an emulation system that would allow the user complete freedom to develop new systems' architecture as he saw fit to provide new development system capability that currently did not exist in the market.

The Slave Emulator is unbundled in the respect that the emulation processor, emulation memory, and emulator bus are completely separated from the development systems' master control computer. **(Fig. 1)** Emulation is truly transparent and no processor resource restrictions are placed on the user's system. Full real-time emulation is started just as soon as the Slave Emulator power switch is turned on, and from that time on the emulator acts as if it were the processor chip itself. Operation of the user's system is not halted or otherwise interfaced with by the emulator unless specifically told to do so by the user at the console of the master control computer. For normal debugging requests only the absolute minimum of emulation processor time is used to answer the user's requests.

The master control computer for the Slave Emulator is the GenRad/Futuredata 2300 Advanced Development System (ADS) computer. This unit is the same console used to support the existing line of bundled emulators and can easily be upgraded to support the new Slave Emulator concept.

The 2300 ADS provides all the software development resources and Slave Emulator control programs needed to complete the user's development system. It can have its own stand-alone disk and printer resources or be attached to the GenRad/Futuredata 2301 Network Control Processor (NCP) in order to share disk and printer resources with a network configuration.

The new Multi-emulator Debugger software running on the 2300 ADS with Slave Emulators allows a single user to control and debug a system consisting of up to eight μPs running simultaneously. The debugger uses a concise command language designed to

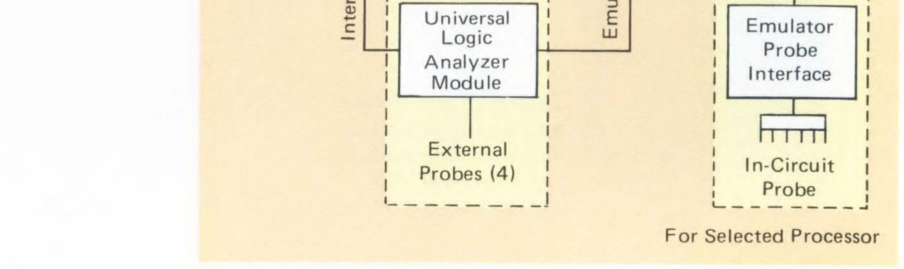

230 ADS Master Control Computer
RS-232
To DS
Interface Processor (Z-80)
SECU ROM/ RAM
Interface Bus
Emulator Control and Status Buffers
Universal Logic Analyzer Module
External Probes (4)
Memory Map Interface
Emulator Bus
Emulator Memory (512K Bytes Max.)
Emulator Personality Module
Emulator Probe Interface
In-Circuit Probe
For Selected Processor

Fig 1 Slave emulator block diagram.

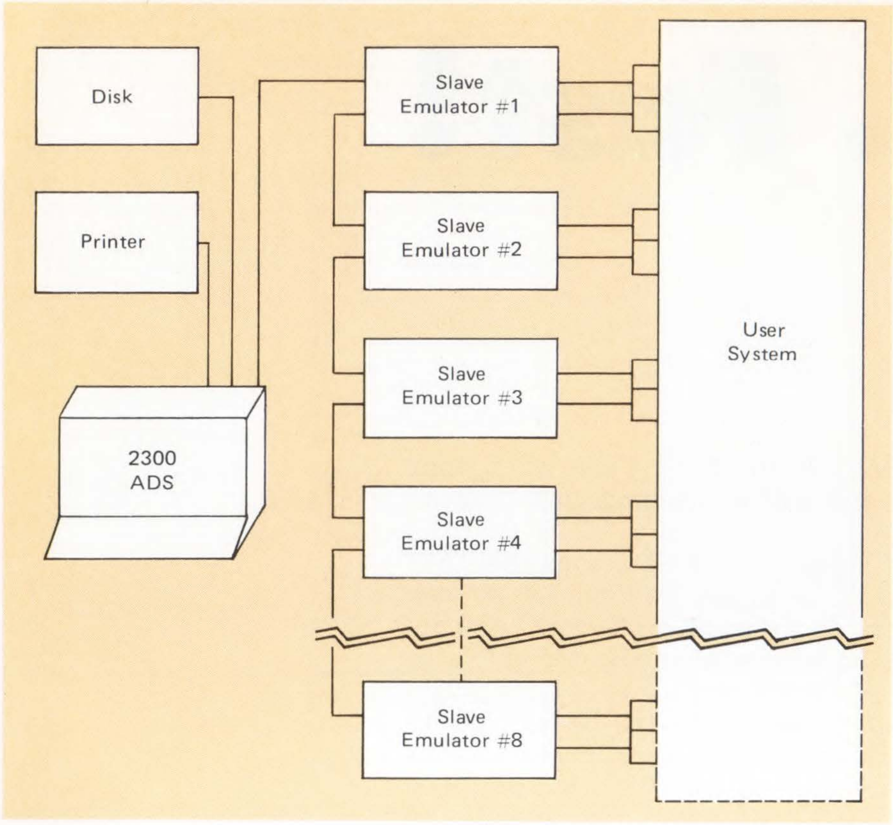


Fig 2 System diagram multiple slave emulators

maximize usability by minimizing keystrokes and utilizing confirmation of critical commands.

The 2300 ADS software set also provides a file manager, text editor, compilers, assemblers, linker and utilities. The 2302 Slave Emulator includes a logic analyzer and is equipped with the desired processor personality modules. If multiple Slave Emulators are utilized, the units are linked together via a high-speed RS232 daisy chain **(Fig. 2)**. Together, master computer, the Multi-emulator Debugger, and the Slave Emulator provide a new level of capability for the development and test of multi-processor systems.

It should be emphasized here that μP-based products are no longer limited to the use of just one processor. Selecting the right processor for a particular function remains a challenge, particularly since so many options exist. The process of checking and optimizing each selection is greatly aided by the slave emulation concept; however, in many cases, the concept of simultaneous emulation of multiple chips is an absolute necessity.

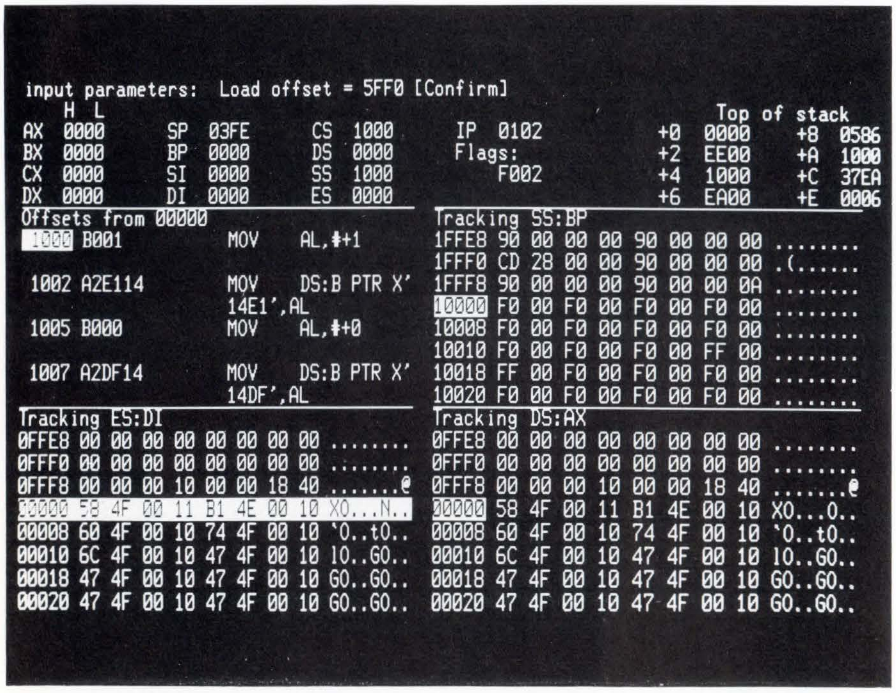
With this new system, the user is allowed to interact with any of the slave emulators at any time via the keyboard, the CRT and the emulator debugger. The user can switch debugging operations to any slave emulator by use of the debugger context switch command. With the CRT the user can display up to four memory areas and CPU of any one processor system **(Fig. 3)**. The CRT is also used to display error messages and user prompts. The user interface is further enhanced by the use of reverse and double intensity video effects on the CRT display.

The Slave Emulator thus provides real-time, full-speed, transparent emulation. This remains true even when all memory resources are mapped into the slave emulator memory, and for processors with memory cycle times down to 100ns. The Slave Emulator requires no CPU resources in the target system such as memory space, I/O space or interrupt vectors. Up to 128K bytes static RAM or 512K bytes dynamic RAM can be mapped in 4K or 16K blocks to any locations within a total 1M byte memory space. Mapping of memory references to internal or external memory is done in 256 byte sections. Write protection throughout the 1M byte space is also done on a 256 byte basis. All I/O is mapped externally.

As noted previously, the Slave Emulator also provides a 64 channel logic analyzer. Sixty bits of information are acquired from the internal bus and four bits from external probes. Examination of the logic analyzer trace and logic analyzer setup are integral functions of the debugger. Four hardware breakpoints, settable for trigger combinations and sequences are provided to control trace and debug operations. An example of a slave emulator logic analyzer display for the 8085 is shown in **Fig 4**.

Non-stop visibility

Perhaps the most significant feature of the slave emulator is that it allows debugging operations to be performed concurrently with the normal operation of the system under test. This is



```
input parameters:  Load offset = 5FF0 [Confirm]
    H  L                                                         Top of stack
AX  0000       SP  03FE       CS  1000     IP  0102          +0  0000    +8  0586
BX  0000       BP  0000       DS  0000     Flags:            +2  EE00    +A  1000
CX  0000       SI  0000       SS  1000         F002          +4  1000    +C  37EA
DX  0000       DI  0000       ES  0000                       +6  EA00    +E  0006
Offsets from 00000                         Tracking SS:BP
 1000 B001             MOV    AL,#+1       1FFE8 90 00 00 00 90 00 00 00 ........
                                           1FFF0 CD 28 00 00 90 00 00 00 .(......
 1002 A2E114           MOV    DS:B PTR X'  1FFF8 90 00 00 00 90 00 00 0A ........
                       14E1',AL            10000 F0 00 F0 00 F0 00 F0 00 ........
 1005 B000             MOV    AL,#+0       10008 F0 00 F0 00 F0 00 F0 00 ........
                                           10010 F0 00 F0 00 F0 00 FF 00 ........
 1007 A2DF14           MOV    DS:B PTR X'  10018 FF 00 F0 00 F0 00 F0 00 ........
                       14DF',AL            10020 F0 00 F0 00 F0 00 F0 00 ........
Tracking ES:DI                             Tracking DS:AX
0FFE8 00 00 00 00 00 00 00 00 ........     0FFE8 00 00 00 00 00 00 00 00 ........
0FFF0 00 00 00 00 00 00 00 00 ........     0FFF0 00 00 00 00 00 00 00 00 ........
0FFF8 00 00 00 10 00 00 18 40 .......@     0FFF8 00 00 00 10 00 00 18 40 .......@
00000 58 4F 00 11 B1 4E 00 10 XO...N..     00000 58 4F 00 11 B1 4E 00 10 XO...O..
00008 60 4F 00 10 74 4F 00 10 `O..tO..     00008 60 4F 00 10 74 4F 00 10 `O..tO..
00010 6C 4F 00 10 47 4F 00 10 lO..GO..     00010 6C 4F 00 10 47 4F 00 10 lO..GO..
00018 47 4F 00 10 47 4F 00 10 GO..GO..     00018 47 4F 00 10 47 4F 00 10 GO..GO..
00020 47 4F 00 10 47 4F 00 10 GO..GO..     00020 47 4F 00 10 47 4F 00 10 GO..GO..
```

Fig 3 8086 Multi-Emulator Debugger Display showing the 8086 register display along with an image of the top of the user's stack.

accomplished by allowing the emulation processor to return immediately to the target system program after completing a short debug sequence. No longer must the system under test be halted for relatively long periods of time, making debugging of time-critical processes and real-time systems impossible.

The internal architecture of the slave emulator affords, to the sophisticated user, the ability to specify special debug sequences to be executed by the emulation processor. This allows the user to develop a command set tailored specifically to the problems being attacked.

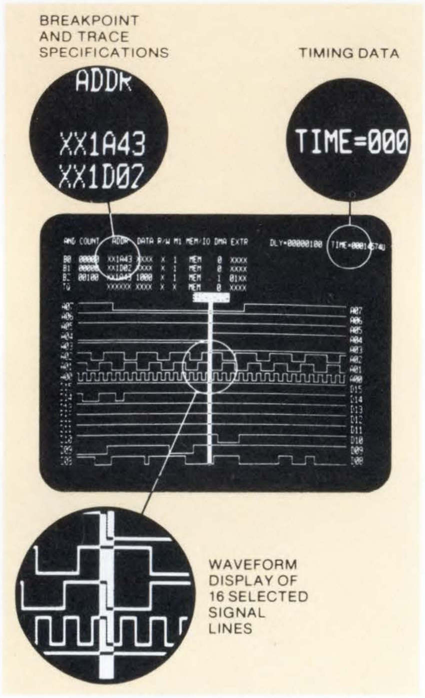

BREAKPOINT AND TRACE SPECIFICATIONS
ADDR
XX1A43
XX1D02
TIMING DATA
TIME=000
WAVEFORM DISPLAY OF 16 SELECTED SIGNAL LINES

Fig 4 Slave emulator logic analyzer display.

Another unique capability of the slave emulator is its ability to allow the user to stimulate the system under test. Without halting normal system operation, variables can be forced out of range, or I/O ports can be manipulated. This is possible because of the truly transparent nature of the emulation provided by the slave emulator system.

The slave emulator along with the command file processing function of the Multi-emulator Debugger provide the ability to do automated testing or production testing of single or multiple processor systems. Tests can be run on read/write memory, read only memory, input/output, interrupts, and even multiprocessor interaction. Tests can be fully automated or set up to allow operator intervention using the CRT and keyboard of the ADS.

Real-time environment

An example of using the Slave Emulator in testing in a real-time system would be in the development of a control system for automotive engine. Here the engine control processor is required to constantly maintain data on temperatures, pressures, throttle position, RPM and rotational position. At the same time, the processor must make calculations and provide precise ignition timing, control of fuel injection and emission control. Since the processor must be running in order for the engine to be running, this is a class of real-time system. The 2302 Slave Emulator has been designed to provide the developers of these types of systems with a real-time debugging tool that will allow them a window into the actual operating real-time system.

In the case of the engine control processor, utilizing the Slave Emulator's ability to perform tests very rapidly and transparently allows tests to be performed while the automotive engine is actually running. This prevents the constant restarting or resetting up of the conditions which are being examined, as would be necessary if the control processor and engine were halted each time the user examined some particular element.

Multi-processor environment

Next, consider a network of point-of-sale terminals connected to a central computer. This network can be examined and debugged by emulating with a Slave Emulator each of the processors on the network which is involved in any particular problem. Up to eight processors of any supported type can be debugged simultaneously using the Multi-Emulator Debugger. While interacting with any particular Slave Emulator, the user has the full capability of the debugger and Slave Emulator at hand. To debug or test the network of point-of-sale terminals, a breakpoint can be set in any processor under test to detect the particular condition desired. This breakpoint can be used to trigger the debugging and tracing functions on all processors which are being emulated and allow analysis of complex interactions between various processors of the multi-processor system.

Want additional information? Contact **GenRad/Futuredata**, 6151 West Century Blvd., Suite 1124, Los Angeles, CA 90045, (213) 641-7200.

Rate this article: circle 3L, 3M or 3H on Reader Inquiry Card.