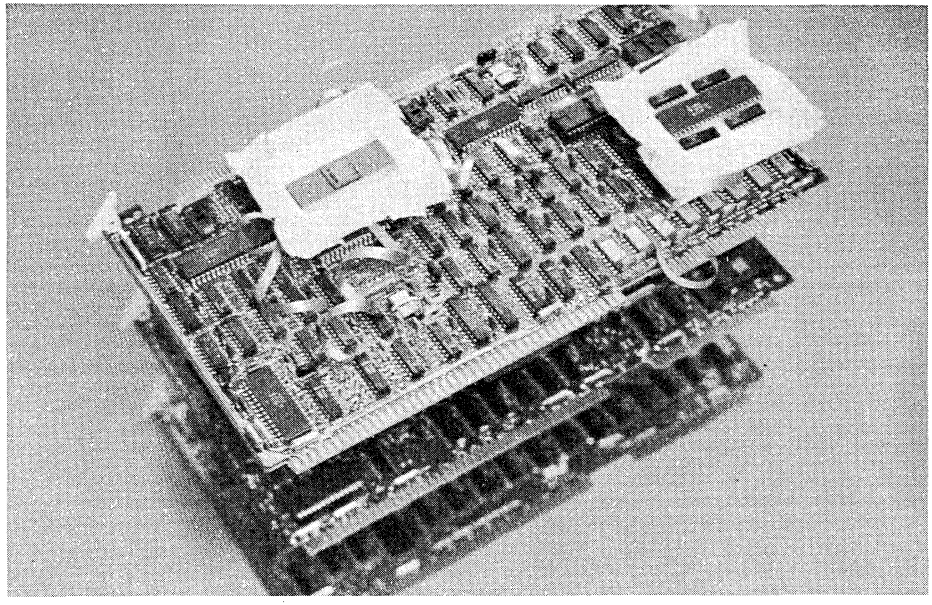
September, 1978

# Triple-bus architecture on a single-board microcomputer

By Jim Johnson, Craig Kinnie and Mike Maerz

Electronic Design 15/ July 19, 1978

# **Triple-bus architecture** lets a single-board microcomputer's CPU operate at full speed while other system components share the main memory.



The introduction of Intel's iSBC 80/30 marks the beginning of the third generation of single board computer architecture. Two features separate the new microcomputer from second-generation single-board $\mu$Cs. The major one is a triple-bus architecture that supports a dual-port memory. As a result, the on-board CPU does not tie up the main system bus (Intel's Multibus) when using the memory. Moreover, with two ports, the memory becomes a global resource, accessible via the three buses from the on-board 8085A CPU as well as from remote CPUs and other external devices in multimaster schemes.

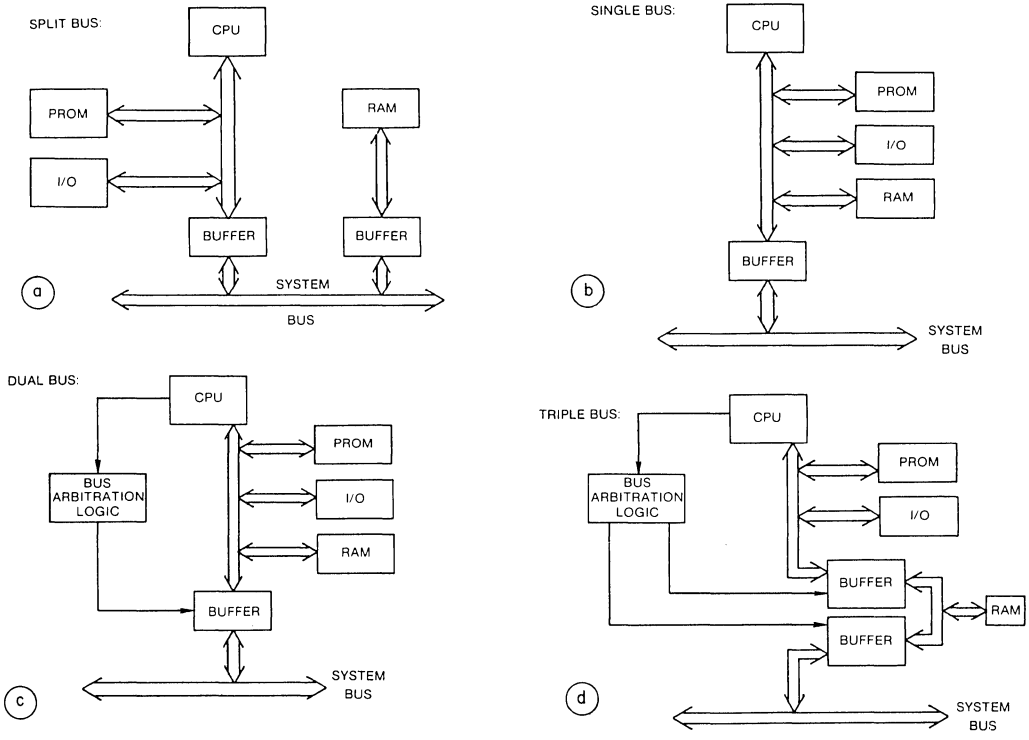
In addition, the 80/30 contains two microprocessors: an 8085A acting as the master CPU and an 8041 single-chip microprocessor acting as a slave, or intelligent-I/O, processor.

**Jim Johnson,** Project Leader, **Craig Kinnie,** Project Manager, and **Mike Maerz,** Marketing Manager, Intel Corp., Santa Clara, CA 95051.

To appreciate the benefits of the 80/30's triple-bus, dual-port memory architecture, examine the following problem. Now that fully one fourth (16-kbytes) of the available memory space in a 64-kbyte $\mu$C system can reside on a single-board $\mu$C, the CPU must share these 16-kbytes with other system components, such as direct-memory-access devices, discs and other processors. What's the best solution—especially when, in many applications, 16-kbytes is all the memory that's required by the whole system?

### Alternatives have problems

The most straightforward way is a split-bus architecture, in which both the CPU and the system have equal access to the memory (Fig. 1a). While the system bus will be able to handle memory access efficiently from devices tied to it, it will be tied up by the CPU—so external operations not related to memory accesses will be hindered.




1. **Microcomputer-bus organizations** takes several forms: In a split-bus approach (a) the CPU and system have equal access to memory, but the CPU ties up the system bus; in a single-bus (b), the CPU encounters extra delays in using the system bus. A dual-bus structure (c) also has buffer delays, and no system access to on-board memory. But a triple-bus (d) avoids all these problems, allowing total system access to memory.

A single-bus approach (Fig. 1b) is hampered by buffer and bus-intervention delays which limit the CPU's performance. And dual-bus architecture (Fig. 1c), while granting the CPU exclusive access, does not allow other bus masters access to the memory. Also dual-bus suffers from buffer delays.

A triple-bus, dual-port architecture (Fig. 1d) provides the benefit of both single and dual-bus architectures: total system access and exclusive access by the CPU. But it also has its disadvantages: Dual-port architecture requires many buffers as well as access-arbitration logic. However, 20-pin octal buffers introduced by several manufacturers don't take up nearly as much board space or cost as much as equivalent standard buffers. Since the octal buffers come in unidirectional or bidirectional forms—and at nearly the same cost—the three-bus approach used on the 80/30 actually takes only as many packages as the split-bus approach.

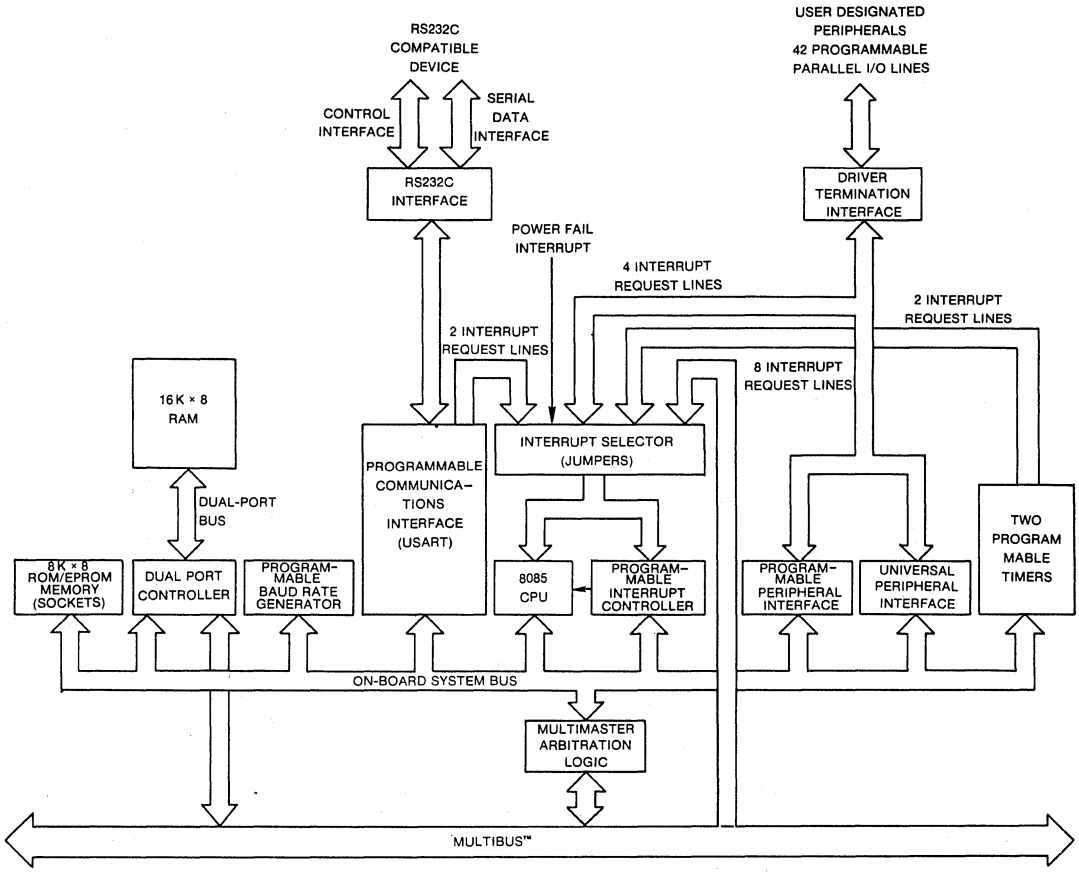
Access arbitration is solved in the 80/30 with cycle status signals from the 8085A CPU. Instead of providing equal access to the RAM from both the CPU and the system, the arbitration logic is designed to favor the CPU. By assigning the default state of the arbiter to the CPU, the logic anticipates a CPU memory access and reserves the memory until the cycle is complete.

In addition, if an on-board CPU access is imminent, a reservation signal derived from the 8085A CPU status signals, the ALE (address latch enable), the address, and the cycle status signals (SO, SI, IO/$\overline{M}$) will hold off bus contention. As a result, the CPU can operate at full speed without tying up the system bus.

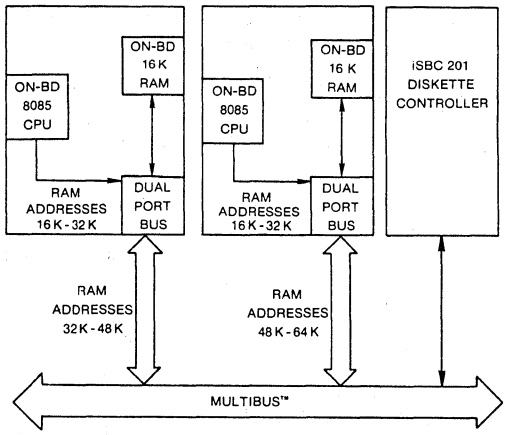
Of course, this extra CPU performance cuts into the rest of the system's memory-access time. However, the penalty imposed by the arbiter is less than 200 ns—less than the time it would take a DMA device to regain control of the bus in the split-bus approach, where access must be interleaved.

## A bus hierarchy

The three buses in the 80/30 hierarchy (Fig. 2) are an on-board bus, a dual-port (DP) bus and the Multibus (system bus). Innermost is the on-board bus, which connects the 8085A, all on-board I/O peripherals and ROM. The next bus in the hierarchy, the dual-port connects a dual-port controller, 16-kbytes of dynamic RAM and a dynamic RAM controller. The




2. **The full 80/30 one-board microcomputer** is organized around its three buses: on board, dual-port, and the external-system Multibus. The main CPU, an 8085A, runs at 2.76 MHz, while an 8041A one-chip microprocessor serves as a peripheral controller or slave processor, running with a 2.6-ms cycle time.




3. **The microcomputer's on-board memory** may be addressed independently by the on-board central processor and Multibus bus masters to increase the efficiency of usage of the total available memory space.

outermost bus, the Multibus, offers modules that permit either the expansion or addition of system resources.

With the on-board bus, the 8085A communicates with its on-board I/O and ROM (or PROM, if desirable) and the dual-port bus. Since the on-board bus permits access to the I/O and ROM only from the 8085A, all I/O and ROM (up to 8-kbytes are the 8085A's private property). And as a result, the 80/30 can operate on its on-board bus while another Multibus master uses the Multibus, accessing data from the board's dual-port RAM without reducing processor speed.

The dual-port (DP) bus contains 16-k of read/write memory, implemented with Intel's 2117 16-kbyte dynamic RAM and the 8202 dynamic RAM controller (DRC). The DRC interfaces the DP bus to the 16-kbytes of dynamic RAM, and provides an almost static-RAM type interface. It provides the system with multiplexed addresses, address strobes, and refresh control to the RAM, as well as refresh/access arbitration and acknowledges.

The RAM on this bus can be accessed from either

the 8085A on the 80/30 or the Multibus. The DP controller arbitrates the RAM requests and performs the bus exchanges.

The DP controller always leaves the DP bus under the control of the 8085A when it is not in use. This permits the 8085A to operate at maximum processor speed when controlling the bus, since there isn't any bus-exchange overhead. When the Multibus requests access to the DP RAM, the DP controller transfers control of the DP bus to the multibus, as soon as the DP bus is not busy. Once the Multibus transfer is complete, the DP bus is returned to the 8085A.

### Multiple communication

The DP controller has two independent address decoders—one for decoding Multibus requests, the other for 80/30 requests. This not only permits the address space of the memory to be located in two different parts of memory (Fig. 3), it enables several 80/30s to talk to each other over the Multibus, while sharing the same on-board address as seen by the 8085A. Thus, one program can be loaded in any 80/30 without relinking and relocating the software for execution.

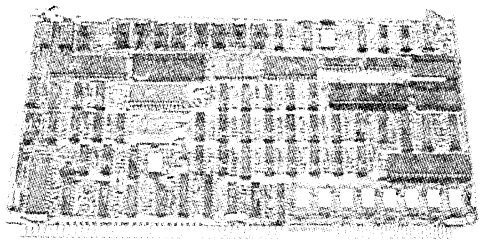
Each bus can communicate either within itself, or with the adjacent bus. Thus, the on-board bus cannot communicate directly with the multibus. However, when the CPU makes a bus request, the on-board and dual-port buses simultaneously determine if they can fulfill it. If the on-board bus can acknowledge the request, it does so, and the DP bus control is not required to determine if the DP bus can acknowledge the request. If the DP bus, not the on-board, can acknowledge the request, it does so, and the controller then lets the CPU use the bus. Thereafter, the RAM controller completes the operation and generates an acknowledge signal.

If neither the on-board nor DP bus can fill the bill, the Multibus is solicited by the CPU. Since a bus can only communicate with an adjacent bus, the on-board bus must request the DP bus to communicate with the Multibus via the DP controller. The on-board bus will retake control of the DP bus only after the request to use the Multibus is granted. This prevents lockout problems with the DP bus, where the CPU requests the Multibus when it is controlled by another bus master accessing the DP RAM.

How the 80/30 performs is directly related to how many buses it must use to complete a requested operation. The on-board bus always operates at maximum processor speed. The DP bus operates at maximum only if it hasn't been busy and a memory refresh cycle was not in process. The processor speed when the Multibus is used depends on bus overhead involved and the type of module requested.

The 80/30 boasts more than a three-bus architecture. For one thing, its I/O is designed to interface to a wide variety of external devices, including switches, motor drives, bistable sensors, displays, keyboards, printers, teletypewriters, communicator modems, cassettes and other computers. This versatility is provided with LSI programmable devices such as Intel's 8255 programmable parallel I/O device, 8251A programmable communication channel, 8253 interval timer, 8259 interrupt controller, and 8041A/8741A universal peripheral interface (UPI).

## The 80/30 in brief



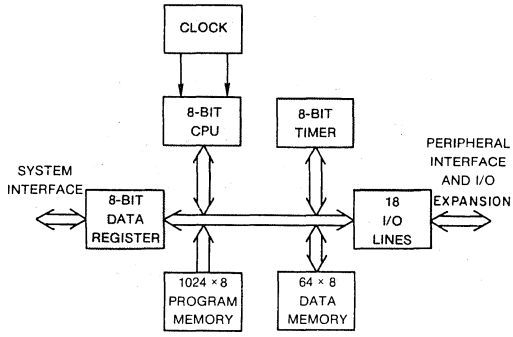
The iSBC 80/30 uses the latest LSI components to obtain the highest performance of any Intel single-board computer. Built on a 6.75 × 12-in. board, it contains the following features:

- 8085A central processor operating at 2.76 MHz.
- 16-kbytes of dual-port RAM using Intel's new 16-kbyte dynamic RAMs and 8202 dynamic RAM controller.
- Sockets for 2, 4 or 8-kbytes of ROM using Intel's 2758, 2708, 2716, or 2332 EPROMs or ROM replacements.
- A socket for Intel's 8041A/8741A universal peripheral interface (UPI) having 18 software-configurable I/O lines with sockets for drivers/terminators.
- A programmable serial-communication channel with RS-232 interface and programmable baud rate.
- Multibus control logic which allows up to 16 masters to share the system bus.
- 12 vectored priority interrupts.
- Two programmable 16-bit BCD or binary internal timers.

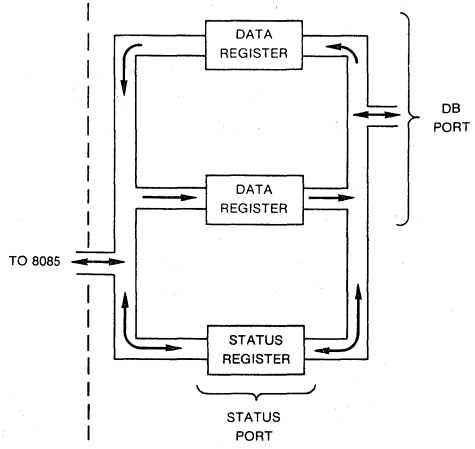
### The slave processor

The ability to interface this wide variety of external devices is facilitated by the 8041A/8741A UPI (Fig. 4), which can be added to the 80/30. The UPI is a complete single-chip microcomputer which acts as a peripheral to the 8085A. It is completely user-programmable with 1-kbyte of ROM (8041A) or EPROM (8741A) memory for data storage. The UPI allows you to fully specify your control algorithm in the peripheral chip without relying on the 8085A. Devices such as printer controllers and keyboard scanners can be completely self-contained, relying on the 8085A only for data transfers.

The UPI is a powerful 8-bit CPU with a 2.6-ms cycle time and an instruction set optimized for bit manipu-




4. **The 8041A/8741A single-chip microcomputer** (UPI-41) has its own on-chip ROM and RAM and can be programmed to perform various peripheral control functions.




5. **The UPI's two data registers** are organized so that the 8085A CPU can write in just one register and read from the other. As a result, the two registers appear as one register to the main 8085A CPU.

lation and I/O operations. It contains an 8-bit counter/timer, buffers to communicate with the 8085A, and two 8-bit programmable I/O ports, which can be customized by software or by plugging in suitable line drivers or terminators into sockets. The UPI also has two input bits that it can test directly. An RS-232 driver and receiver on the 80/30 permit the UPI to be programmed as a simple serial-communication channel.

### Interfacing to the on-board bus

The UPI interfaces asynchronously with the on-board bus using two data and two status registers. The UPI's two internal data registers appear to the 8085A as only one register, since one data register can be written into only by the UPI and read only by the 8085A, and the other can be written into only by the 8085A and read by the UPI (Fig. 5). This is done to prevent the two CPUs from simultaneously writing into a data register.

The UPI can communicate with the 8085A by loading a data register and then returning to its previous control task. The 8085A can periodically poll the UPI status port for the valid-read (VR) flag, which is set in hardware when the UPI writes to its data port, or the UPI can generate an interrupt to the 8085A via an I/O bit that can be programmed to be the VR flag.

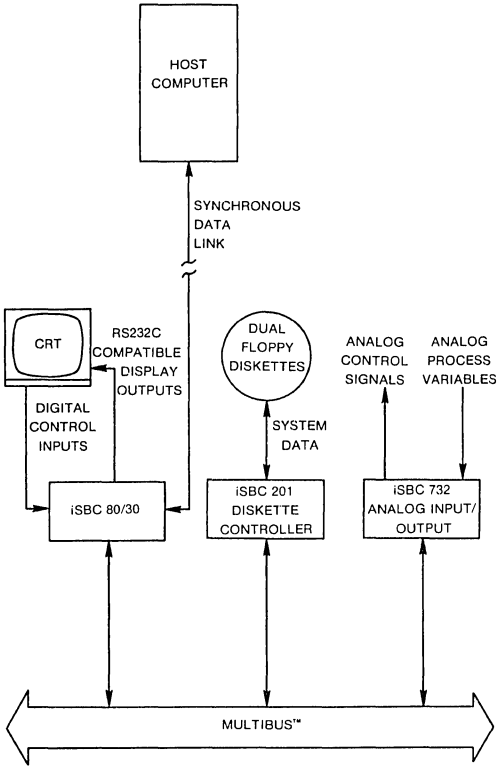
Once the 8085A determines the VR flag is true, it can transfer the data to its own memory without disturbing the UPI. The VR flag is automatically cleared after the data are transferred. Similarly, when the 8085A transfers data to the UPI, a valid output (VO) flag is set and an interrupt to the UPI is generated (if enabled) automatically. Once the UPI transfers the data, the VO flag is cleared. The VO flag can also be programmed to a port bit for generating interrupts to the 8085A to indicate that the transfer is complete.

### An extensive interrupt system

The 80/30 provides 12 vectored priority interrupts, four of which are handled directly by the 8085A's interrupt-processing capability and routed to fixed, unique memory locations. The remaining eight levels are handled via the 8259A programmable interrupt controller (PIC), which generates a unique memory address for each level. These addresses are equally spaced at intervals of four or eight (software-selectable) bytes. This 32 or 64-byte block may be located to begin at any 32 or 64-byte boundary in the 65,536-byte memory space. A single 8085A jump instruction at each of these addresses then provides the linkage to locate each interrupt-service routine independently anywhere in memory. The PIC provides a selection of four priority algorithms so that the manner in which real-time requests are processed may be configured to meet the requirements of the system under design.

The 80/30 also has two 8253-based programmable 16-bit BCD and binary timers/event counters, which can be used for a variety of functions. Both timers may be set to act as a rate generator (divide-by-N counter), a square-wave generator, a programmable retriggerable one-shot, or one of the timers can be jumper-selected as an event counter. In addition, an interrupt can be generated when a time interval has expired or when a specified number of events has occurred.

To see how useful the 80/30 can be, consider a supervisory control/monitoring system (Fig. 6) using an Intel iSBC 80/30 single-board computer, iSBC 201 diskette controller, and iSBC 732 analog input/output




6. **In this application example,** the 80/30 forms the heart of a remote data-acquisition system. By taking advantage of the one-board microcomputer's dual-port memory and universal peripheral interface, the system achieves a combination of attractive cost and efficiency.

board. Here local commands and process-status signals are given and displayed on a CRT, which is interfaced via the iSBC 80/30 resident UPI and RS232C components. Process variables are converted from analog to digital using the analog I/O board. Control variables are passed over the Multibus from the 80/30 to the 732, where they are converted from digital to analog.

System data are logged on two diskettes, which are controlled by the 201. The controller board's on-board DMA interface accesses the 80/30's dual-port memory and stores the data on one of the floppy discs.

At the end of the day, a remote host processor, interfaced to the 80/30 via a modem (through the 80/30's 8251A and RC232C circuits) can request all or part of the diskette-resident data. Here, the 80/30 uses its on-board dual-port memory as a data buffer for transfers to the host.

Intel's RMX/80 real time executive, disc-file system and analog drivers provide the majority of the system's software.■■

*Note: Multibus and iSBC are registered trademarks of Intel Corp.*