PC-BASED TEST BENCH

In this series of articles we'll be building various PC-controlled test equipment—but first we need a universal interface card.

STEVE WOLFE

A PC IS THE PERFECT THING to use to accumulate, manipulate, plot, and store the results of an experiment. PC-based test equipment has an advantage over traditional instruments: since various instruments share the same PC, the money that would normally be spent duplicating the display, keyboard, etc., can be saved. That's the idea behind this series of articles. We'll build a number of PC-based test instruments, including a capacitance meter, a 100-MHz frequency counter, a logic IC tester/identifier, and an oscilloscope. We'll start this month with an interface card.

The search for the perfect PC interface begins with the serial port. Unfortunately, the serial port is too slow for transferring large quantities of data needed to control and monitor test equipment. Another possibility is the parallel port which can transfer 8 bits in 500 nanoseconds (best case). Unfortunately, the parallel port is not truly bidirectional. A couple of handshake lines can be used as data inputs, but that means converting fast parallel data into slower serial data. Also, several data lines would have to be sacrificed so that they could be used as address lines. Another possible solution would be to connect a circuit directly to the computer's expansion bus. That would be very fast and easy to program, but it would require giving up an expansion slot every time you added another device.

What's needed is a general-purpose, fully bidirectional parallel port that can select and drive different peripherals all connected to a single generic ribbon cable. That is all contained in the I1000 Data Interface that we'll build this month. The I1000 can address up to 256 peripheral devices, all connected in parallel, using 25-conductor ribbon cable. The I1000 is simple to program; an "out" or "write" command sends a byte, and an "in" or "read" command receives a byte.

I1000 operation

Each card in a PC has it's own address. That is necessary to ensure that information intended for a certain card is received only by that card, and to ensure that only one card can place data on the bus at a time. Typically, the I1000 is set to address 768 (hex 300)—an address that IBM left available for prototyping. The I1000 can be re-addressed as needed by changing an address DIP switch. As far as software goes, we'll use BASIC due to it's broad popularity, but almost any other language can be used.

Sending a byte

Refer to Fig. 1 for the following example. When the BASIC instruction, "OUT 768,85" is executed, the byte "85" (01010101) is sent to address "768" (where the I1000 resides). The PC expansion bus address lines A5–A9 are attached to the card-address block, along with the $\overline{\text{ADDRESS ENABLE}}$ ($\overline{\text{AEN}}$) line, which indicates that the address data is valid, and the $\overline{\text{WRITE}}$ ($\overline{\text{WR}}$) line, which indicates that an "out" was performed. If the $\overline{\text{AEN}}$ and $\overline{\text{WR}}$ lines are low (logic 0) and the address lines match the DIP switch settings, an 8-bit magnitude comparator in the card-address block changes state (goes low). That tells the I1000 that the CPU has selected it.

The PC's $\overline{\text{WRITE}}$ pulse, in conjunction with the $\overline{\text{ENABLE}}$ pulse from the card-address block, causes the address latch to store the address, and the data latch to store the PC bus data. At that point, the I1000 is finished using the expansion bus, and it places the data, address, and $\overline{\text{SEND}}$ pulse on the interface cable that is going to the peripheral. The $\overline{\text{SEND}}$ pulse is sent along as confirmation that the data and address information is valid. Approximately 750 nanoseconds later, the I1000 sends a 500-nanosecond peripheral $\overline{\text{WRITE}}$ pulse. By the time the $\overline{\text{WRITE}}$ pulse reaches the peripheral, the data, address, and $\overline{\text{SEND}}$ pulses have finished any ringing associated with parallel interfacing. Additionally, each of the signals mentioned are terminated and buffered on the I1000 and at the peripheral. That defeats any error and noise (reflection, bounce, and

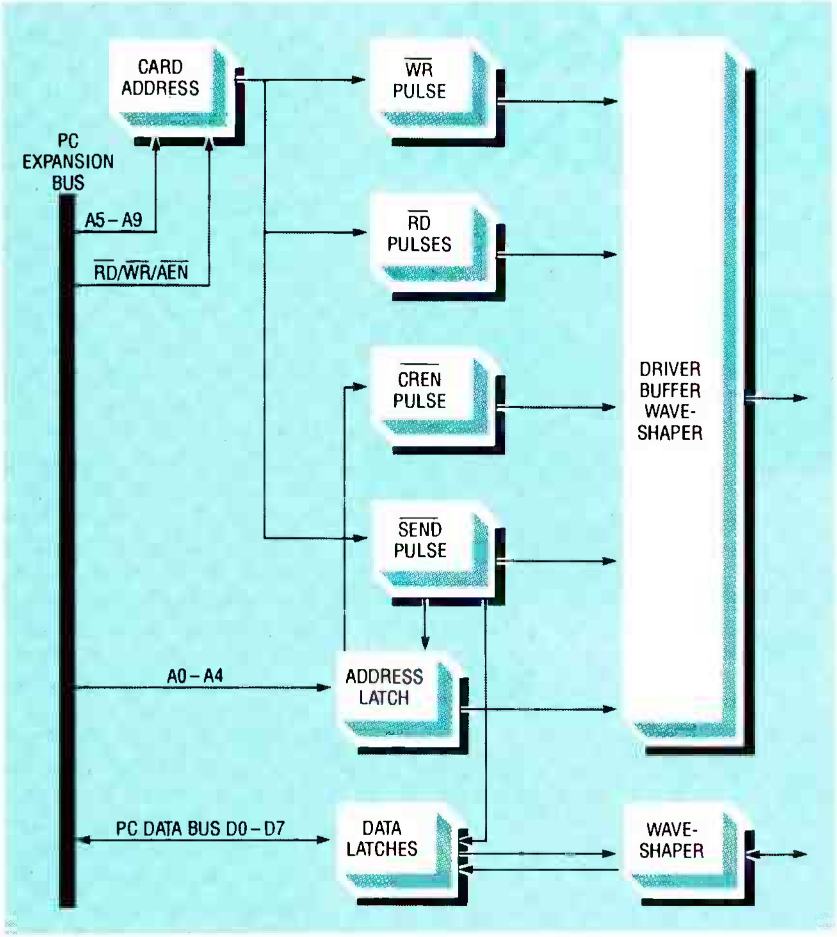


FIG. 1—I1000 BLOCK DIAGRAM. This interface will let your PC communicate with the test equipment we'll be working on in future articles.

crosstalk) problems commonly associated with parallel data transfer. The peripheral responds to the $\overline{\text{WRITE}}$ pulse by storing the data byte (D0–D7) within the location dictated by the address information (A0–A4) that it received.

Receiving a byte

For the following example, we will execute the line of BASIC: A = INP(768) : A = INP(768) : PRINT A. When the $\overline{\text{ADDRESS ENABLE}}$ ($\overline{\text{AEN}}$) and the PC's $\overline{\text{READ}}$ ($\overline{\text{RD}}$) lines are low, the card-address section once again goes low, and the send and address information is sent to the peripheral. A $\overline{\text{READ}}$ pulse is sent to the peripheral 500 nanoseconds later, which causes the peripheral to send the data back to the I1000. The data from the peripheral is stored in the I1000 250 nanoseconds later. The second input statement moves the data from the I1000 to the variable (A). Finally, the byte is displayed on the PC's monitor.

Control register enable

The I1000 has the ability to talk to 32 locations within 256 peripheral devices. That tremendous flexibility is accomplished through the use of the control register. When the I1000 is set to a base address of 768, it is actually active from 768 to 799, and covers 32 addressable bytes. If we say that the variable "bas" is equal to 768, then one I1000 can cover bas + 0 (768) to bas + 31 (799). Within the I1000, bas + 31 has been decoded to a single line. In other words, when an "out" is sent to bas + 31, the $\overline{\text{CREN}}$ line goes low.

When the $\overline{\text{CREN}}$ line goes low, any peripheral attached enters a comparator mode. While in that mode, each peripheral compares the information on the data bus with it's own hardwired identification byte. If they match, that peripheral will attach itself to the data bus. In a peripheral where the bytes do not match, that peripheral will ignore or disconnect itself from the data bus. Once a peripheral has been called, it continues to be connected to the data bus until another bas + 31 activates a different peripheral.

Suppose peripheral 1 is an A/D converter with a unit address of 0 and peripheral 2 is a capacitance meter with a unit address of 4. An "out bas + 31,0" would select the A/D converter unit. The A/D would not actually do anything other than connect to the bus. After that, outs and ins to addresses between bas + 0 (768) and bas + 30 (798) would cause the A/D peripheral to perform it's job. An "out bas + 31,4" at this point would remove the A/D converter from the cable and connect the capacitance meter. Again, outs and ins in the range bas + 0 to bas + 30 would control the instrument selected.

Finally, an "out bas + 31,99" would disconnect both of the peripherals from the interface cable. That occurs because there is no device currently connected with a hard-wired identification byte of 99. The data bus is eight bits wide, so 256 (2^8) different peripherals can be addressed. Leaving bas + 31 for addressing different units, 31 addresses (0–30) remain for accessing IC's within each unit. The total number of locations accessible by one I1000 is 7936 (256 × 31).

Detailed operation

Take a look at the timing diagrams in Figs. 2 and 3 and the schematic in Fig. 4. A 74LS688 8-bit magnitude comparator (IC1) compares DIP switch S1's settings to the address present at address lines A5–A9 (P1, pins A22–A26). It also checks to see that $\overline{\text{WR}}$ and $\overline{\text{AEN}}$ are low. When those conditions are met, IC1 pin 19 goes low, telling the I1000 that it has been selected by the CPU. Address lines A0–A4 (P1 pins A27–A31) are connected to IC10, a 74LS573 address latch. When pin 19 of IC1 goes low, it causes pin 6 of IC2-b (a 74LS86) to go high, latching the address information into IC10. When

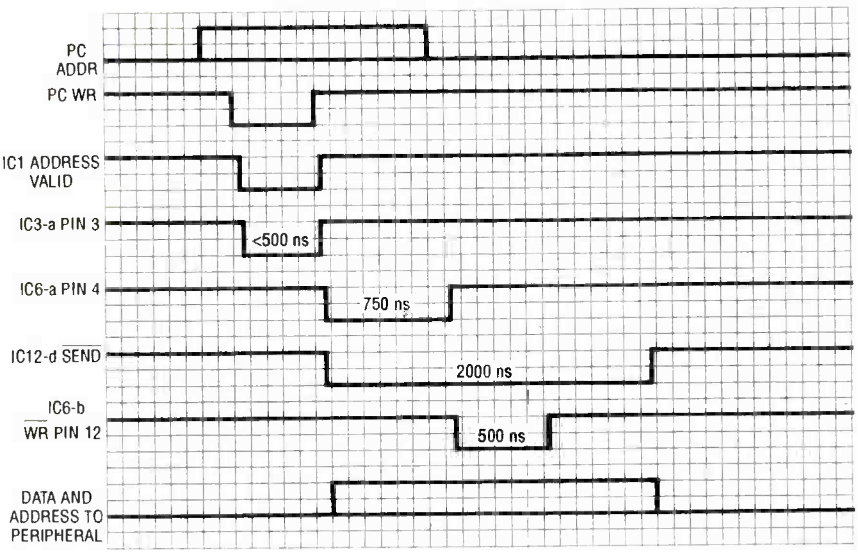

PC ADDR
PC WR
IC1 ADDRESS VALID
IC3-a PIN 3
<500 ns
IC6-a PIN 4
750 ns
IC12-d $\overline{\text{SEND}}$
2000 ns
IC6-b $\overline{\text{WR}}$ PIN 12
500 ns
DATA AND ADDRESS TO PERIPHERAL

FIG. 2—I1000 WRITE, or "out" timing sequence.

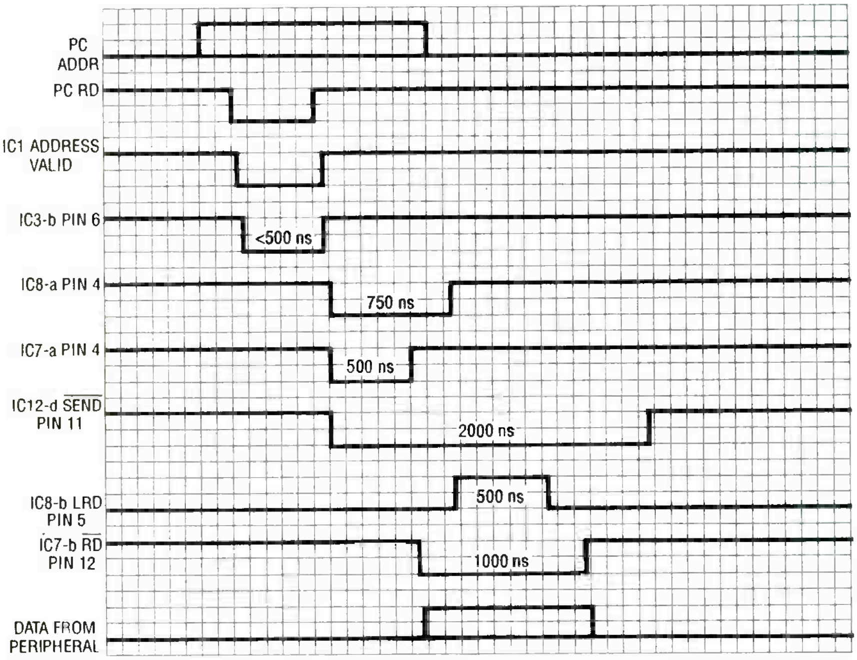


FIG. 3—I1000 READ, or "in" timing sequence.

the $\overline{\text{WR}}$ and $\overline{\text{EN}}$ pulses at the inputs of IC3-a (a 74LS32) go low, the output of IC3-a does the same. That causes the output of IC2-c to go high and moves D0–D7 data from the PC into data latch IC4.

Components IC6–IC9 (74HCT221's) are rising-edge triggered monostable multivibrators (one-shots) triggered by rising pulses. After approximately 500 nanoseconds, the $\overline{\text{WR}}$ and $\overline{\text{EN}}$ pulses return to their inactive high state and, as a result, the output of IC3-a returns to a high state. The rising edge produced by IC3-a triggers IC9-b and IC6-a. The $\overline{\text{WEND}}$ pulse, generated by IC9-a, when ANDed with the $\overline{\text{REND}}$ pulse, produces the $\overline{\text{SEND}}$ pulse. The $\overline{\text{SEND}}$ pulse tells the peripheral that the bus information is valid. The $\overline{\text{WEND}}$ and $\overline{\text{SEND}}$ pulses also enable IC10 and IC4, allowing A0–A4 and D0–D7 onto the peripheral buses.

At the same time IC9-b is triggered, IC6-a is triggered, producing a 750-nanosecond delay pulse. As IC6-a times out, it triggers IC6-b, which produces a 500-nanosecond $\overline{\text{WR}}$ pulse that is centered within the 2-µs $\overline{\text{SEND}}$ timing window. The $\overline{\text{WR}}$ and $\overline{\text{SEND}}$ pulses pass through IC13, a 74LS541 line driver/buffer. The $\overline{\text{WR}}$ pulse is reshaped by R9 and C30 to a waveform more suited to a long cable with inductive reactance. The $\overline{\text{SEND}}$ pulse is similarly reshaped by DIP resistor R10 (pins 6 and 11) and C27. During a $\overline{\text{WR}}$ operation, the data lines D0–D7 are conditioned by R11, R16, and C31–C38 on the way to the peripheral device. The address lines at the output of IC10 (A0–A4) are conditioned by R10 and C22–C26. Those address lines and the $\overline{\text{WEND}}$ pulse are applied to IC11, a 74LS138 demultiplexer. If $\overline{\text{WEND}}$ is low and the address is equal to the base address (768) plus thirty one (as discussed earlier), pin 7 of IC11 goes low producing the $\overline{\text{CREN}}$ pulse.

I1000 PARTS LIST

All resistors are ¼-watt, 1%, unless otherwise noted.
R1, R3, R5—1000 ohms, 5%
R2, R6—4320 ohms
R4—9090 ohms
R7, R8—20,000 ohms
R9—33 ohms
R10, R11—33 ohms, 16-pin DIP resistor
R12–R14—10,000 ohms, multiturn potentiometer
R15—4700 ohms, 10-pin SIP resistor
R16—2200 ohms, 10-pin SIP resistor

Capacitors
C1–C13—0.15 µF, 50 volts, monolythic or polystyrene
C14–C21—105 pF, 100 volts, dipped mica
C22–C29—1500 pF, 63 volts, polystyrene
C30—0.001 µF, 100 volts, ceramic disc
C31–C38—220 pF, 100 volts, ceramic disc
C39—100 µF, 25 volts, electrolytic
C40–C45—10 µF, 35 volts, electrolytic

Semiconductors
IC1—74LS688D 8-bit magnitude comparator
IC2—74LS86D quad 2-input XOR gate
IC3—74LS32D quad 2-input OR gate
IC4, IC5, IC10—74LS573D octal latch
IC6–IC9—74HCT221D dual one shot
IC11—74LS138D demultiplexer
IC12—74LS08D quad 2-input AND gate
IC13—74LS541D octal buffer

Other components
J1—Right-angle PC-mount female DB25 connector
S1—8-position DIP switch
Miscellaneous: I1000 PC board, PC mounting bracket and hardware with DB25 cutout, solder, etc.

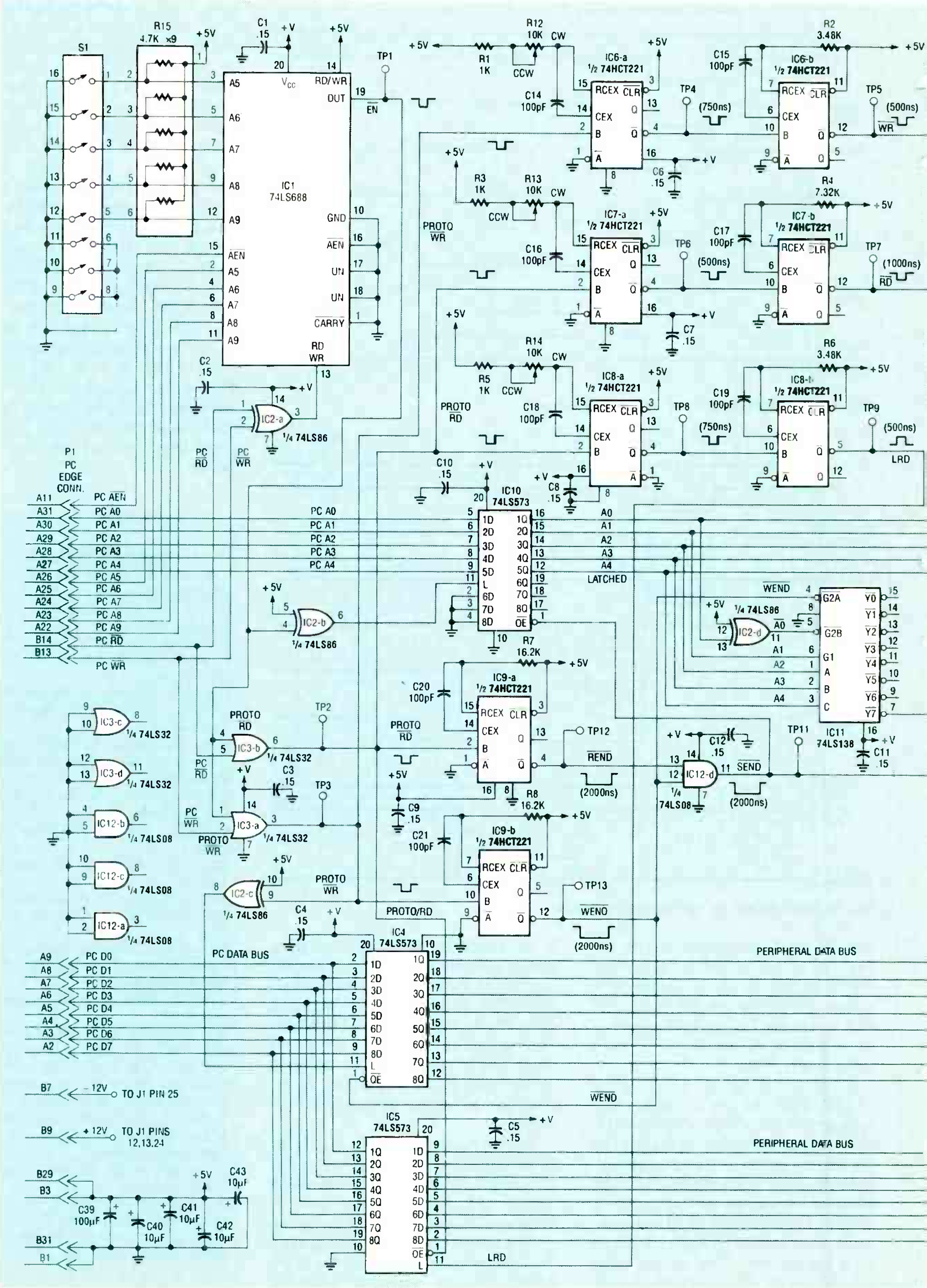
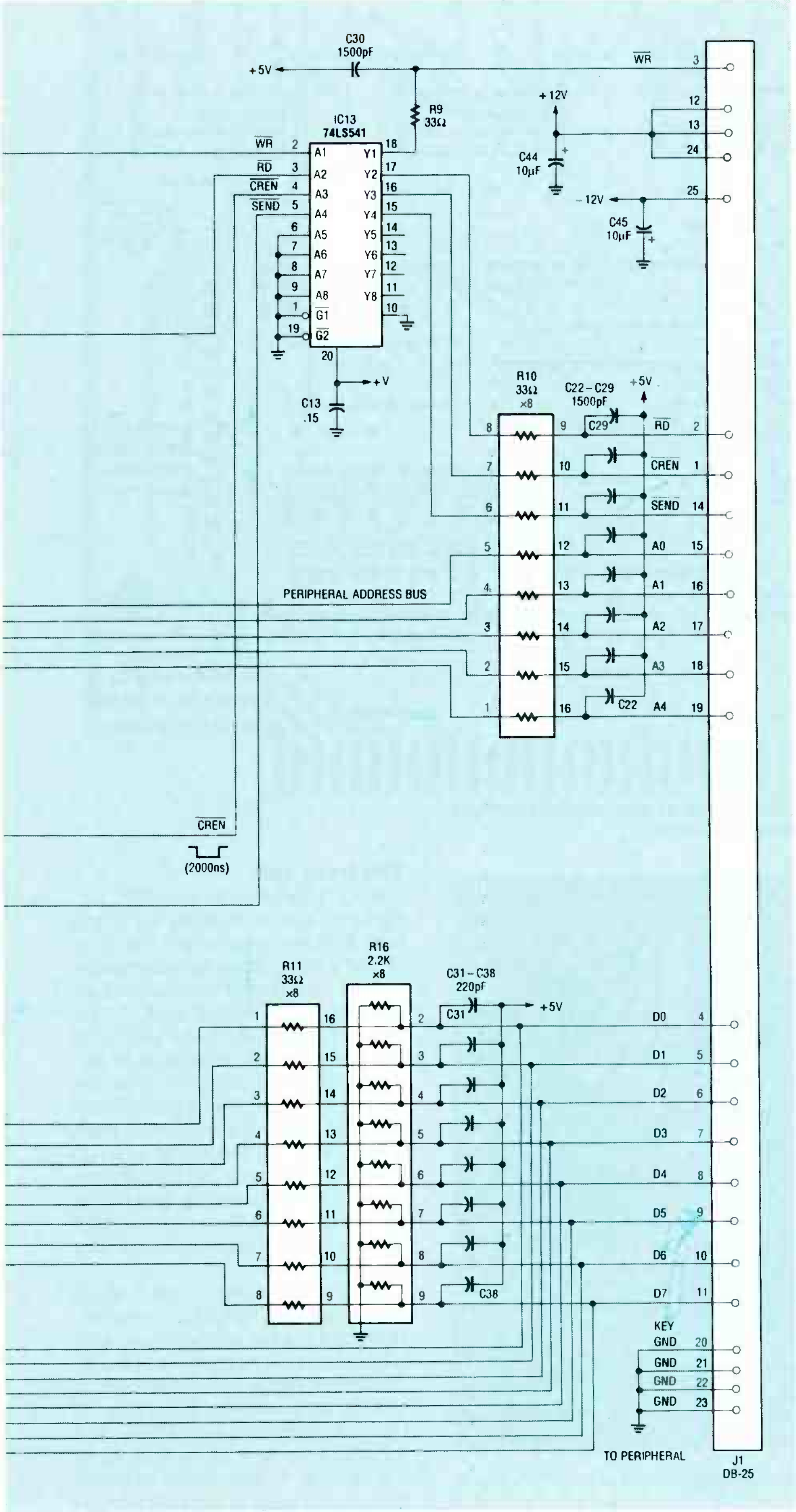


FIG. 4—I1000 SCHEMATIC. The I1000 can talk to 32 locations within 256 peripheral devices to provide tremendous flexibility.



Receiving a byte

When receiving a byte, IC1 operates the same as when it is sending except that the $\overline{\mathrm{RD}}$ line goes low. The address data (A0–A4) is again stored in IC10. The $\overline{\mathrm{RD}}$ and $\overline{\mathrm{EN}}$ pulses go low, and as a result IC3-b transitions low. The PC then reads back the contents of IC5. (The information read back at this point is irrelevant, since information from the peripheral unit has not reached the I1000 yet.) As the $\overline{\mathrm{RD}}$ and $\overline{\mathrm{EN}}$ pulses end, a rising pulse edge occurs at IC3. That activates IC7-a, IC8-a, and IC9-a.

The $\overline{\mathrm{REND}}$ pulse is produced by IC9-a, which, when it passes through IC12-d, becomes $\overline{\mathrm{SEND}}$. A 500-nanosecond delay pulse is produced by IC7-a; as IC7-a times out, it triggers IC7-b, which produces a 1000-nanosecond $\overline{\mathrm{RD}}$ pulse which is sent to the peripheral unit. (The $\overline{\mathrm{SEND}}$ pulse and address information arrived at the peripheral 500 nanoseconds earlier.) Upon receiving the $\overline{\mathrm{RD}}$ pulse, the peripheral sends the $\overline{\mathrm{D0–D7}}$ data to the I1000 (IC8-a went active at the same time as IC7-a, and produced a delay pulse of 750 nanoseconds). As IC8-a times out, it triggers IC8-b to produce a 500-nanosecond latching pulse. The pulse controls the LATCH line of IC5 and stores the information sent by the peripheral during the (still active) 1000-nanosecond $\overline{\mathrm{RD}}$ pulse. A second identical input statement will now cause IC3-b to go low. That again activates IC5 and returns valid data to the PC.

I1000 construction

To build the I1000 interface, you can either buy a PC board from the source mentioned in the Parts List or make one from the foil patterns we've provided. Install parts on the board as shown in Fig. 5. You will notice that for many of the capacitors, there are three holes on the board, with two of them electrically the same. Those two holes are for mounting capacitors of different sizes. Use the pair of holes that best fit the capacitors you use. Figure 6 shows a completed card.

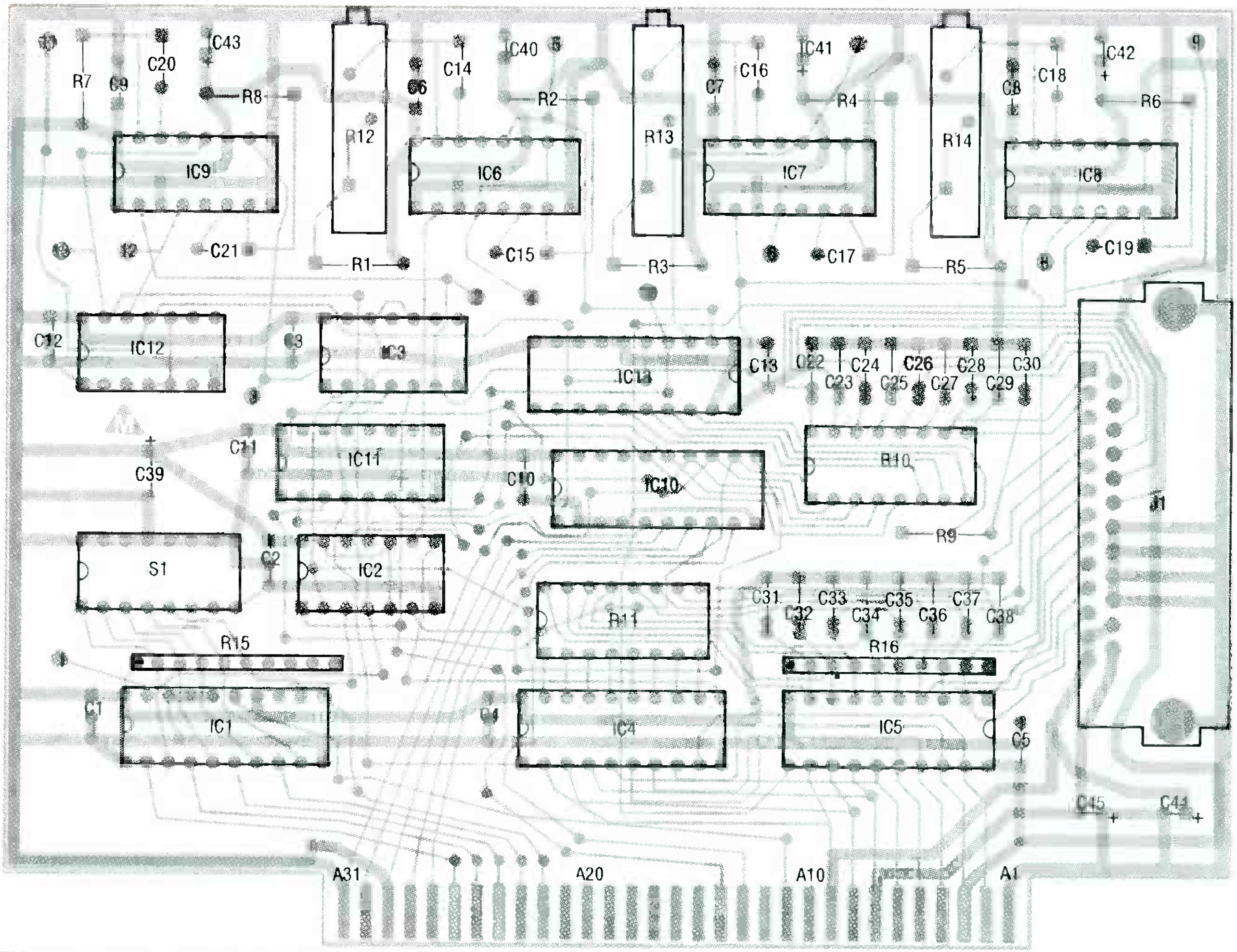


FIG. 5—INSTALL PARTS AS SHOWN HERE. For many of the capacitors there are three mounting holes to accommodate different-sized capacitors.

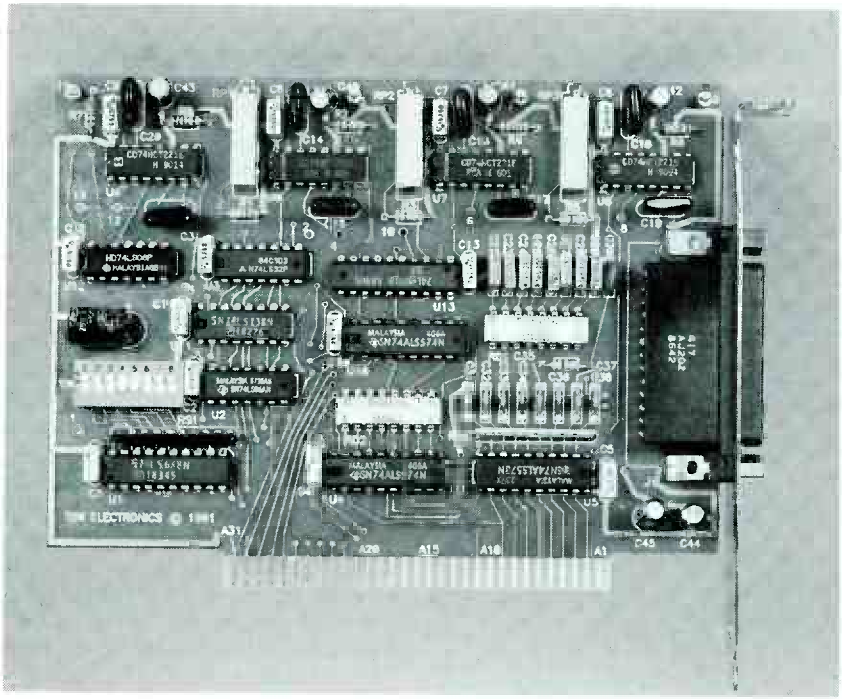


FIG. 6—COMPLETED INTERFACE CARD. This is installed in one of your PC's expansion slots.

The front end

Any I1000-compatible peripheral must contain an interface section to control the flow of data and clean up any noisy pulses. We'll call this interface section the "front end." The front end will be nearly identical for each I1000-compatible peripheral showcased in this series of articles. Each peripheral will contain its own front end, which will be included on the main PC board. Although we will not be discussing any of the PC peripherals this month, let's go over the operation of the front end now.

As shown in Fig. 7, each front end contains a data termination block and an address and handshake termination block. These sections are activated by inserting push-on jumpers. If the jumpers are removed, the termination section will be electrically inert. The I1000 is capable of addressing up to 256 (2^8)

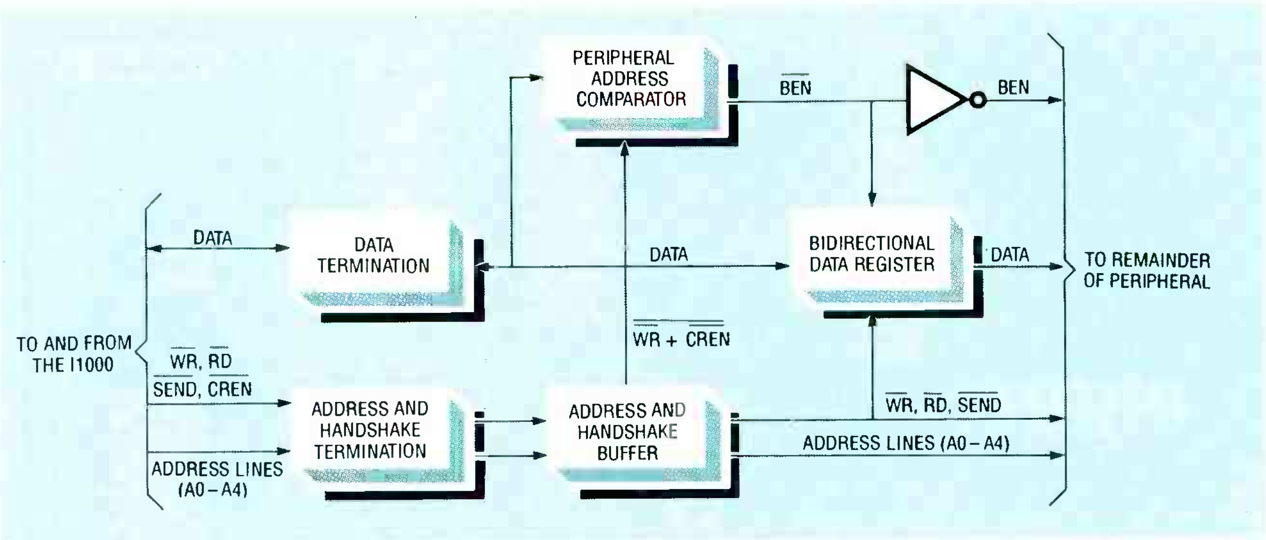
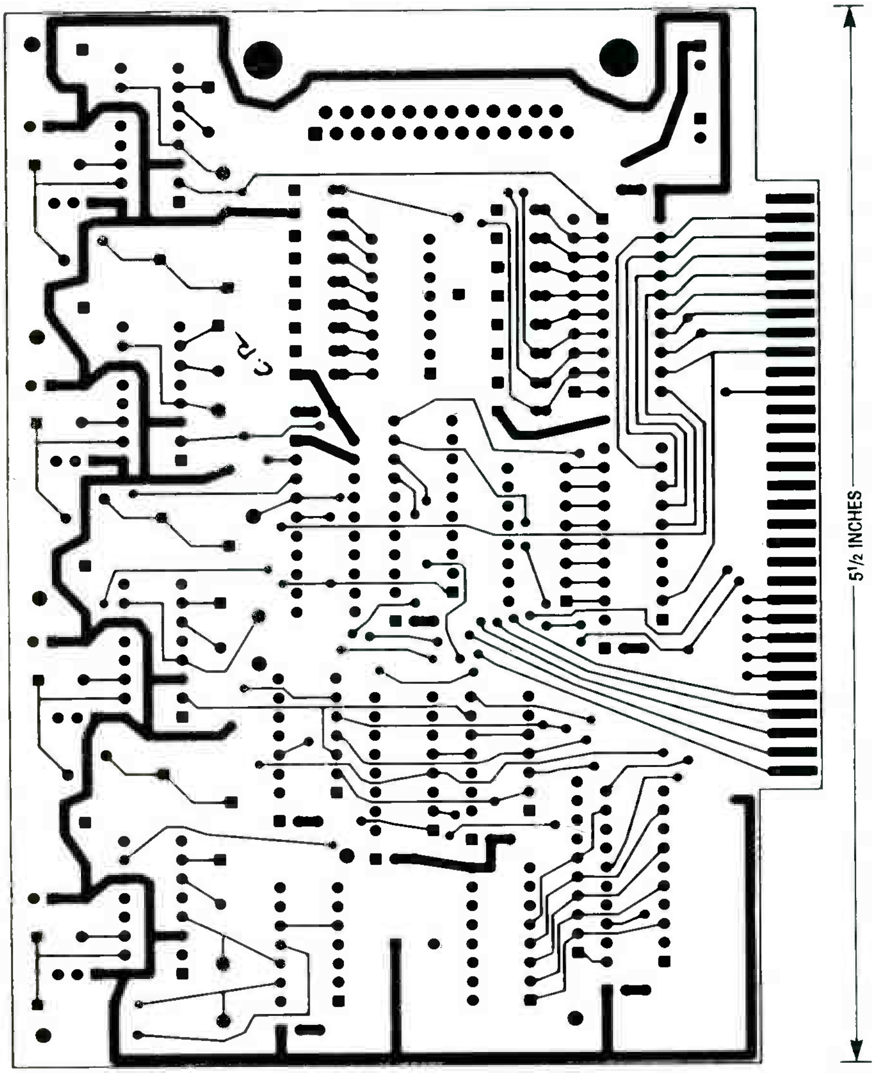

PERIPHERAL ADDRESS COMPARATOR
$\overline{\text{BEN}}$
BEN
DATA
DATA TERMINATION
DATA
BIDIRECTIONAL DATA REGISTER
DATA
TO REMAINDER OF PERIPHERAL
TO AND FROM THE I1000
$\overline{\text{WR}}$, $\overline{\text{RD}}$
SEND, CREN
$\overline{\text{WR}}$ + $\overline{\text{CREN}}$
ADDRESS AND HANDSHAKE TERMINATION
ADDRESS AND HANDSHAKE BUFFER
$\overline{\text{WR}}$, $\overline{\text{RD}}$, $\overline{\text{SEND}}$
ADDRESS LINES (A0 – A4)
ADDRESS LINES (A0 – A4)

FIG. 7—ANY I1000-COMPATIBLE PERIPHERAL must contain an interface section to control the flow of data and clean up any noisy pulses. This front end will be nearly identical for all of the peripherals.



COMPONENT SIDE of the I1000 interface board.

peripherals. The DB-25 connectors on the rear of each peripheral are simply connected in parallel with one another. Termination of the data bus must occur at the most distant point on the bus and only at that point. If more than one peripheral were terminated at the same place, the termination impedance and its location would be altered, thus distorting the performance of the front ends.

After passing the active or inactive termination section, the data bus is attached to the Peripheral Address Comparator (PAC) and the Bidirectional Data Register (BDR). The PAC is responsible for activating a peripheral called by $\overline{\text{CREN}}$ as previously described. Each peripheral's PAC section contains its own unique address. If, during an active $\overline{\text{CREN}}$ pulse, the data on the bus matches the PAC address, the PAC section produces a low $\overline{\text{BOARD ENABLE}}$ handshake ($\overline{\text{BEN}}$). That signal and its complement (BEN) connect the remainder of the peripheral to the data bus and handshake lines ($\overline{\text{RD}}$, $\overline{\text{WR}}$, etc.).

The BDR is now capable of passing data to, and receiving data from, the main peripheral circuitry. The BDR is controlled by $\overline{\text{RD}}$, $\overline{\text{SEND}}$, and $\overline{\text{BEN}}$. Those lines tell the BDR the direction of data movement as well as the timing of that movement. After passing the active or inactive termination section, the address and handshake signals enter the address and handshake buffer. The signals are rounded by the termination sections to minimize crosstalk and other noise associated with fast rise and fall times. The address

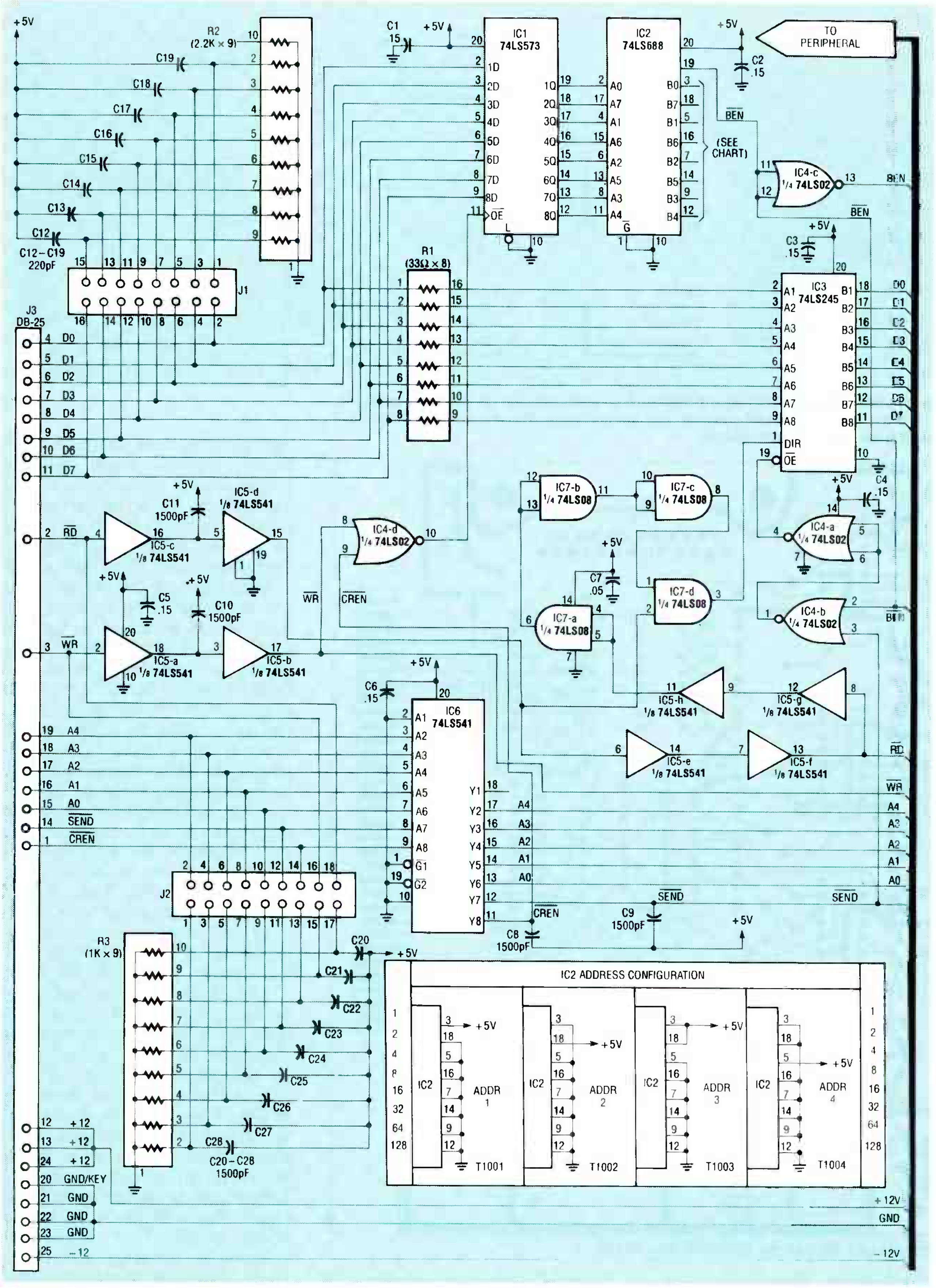
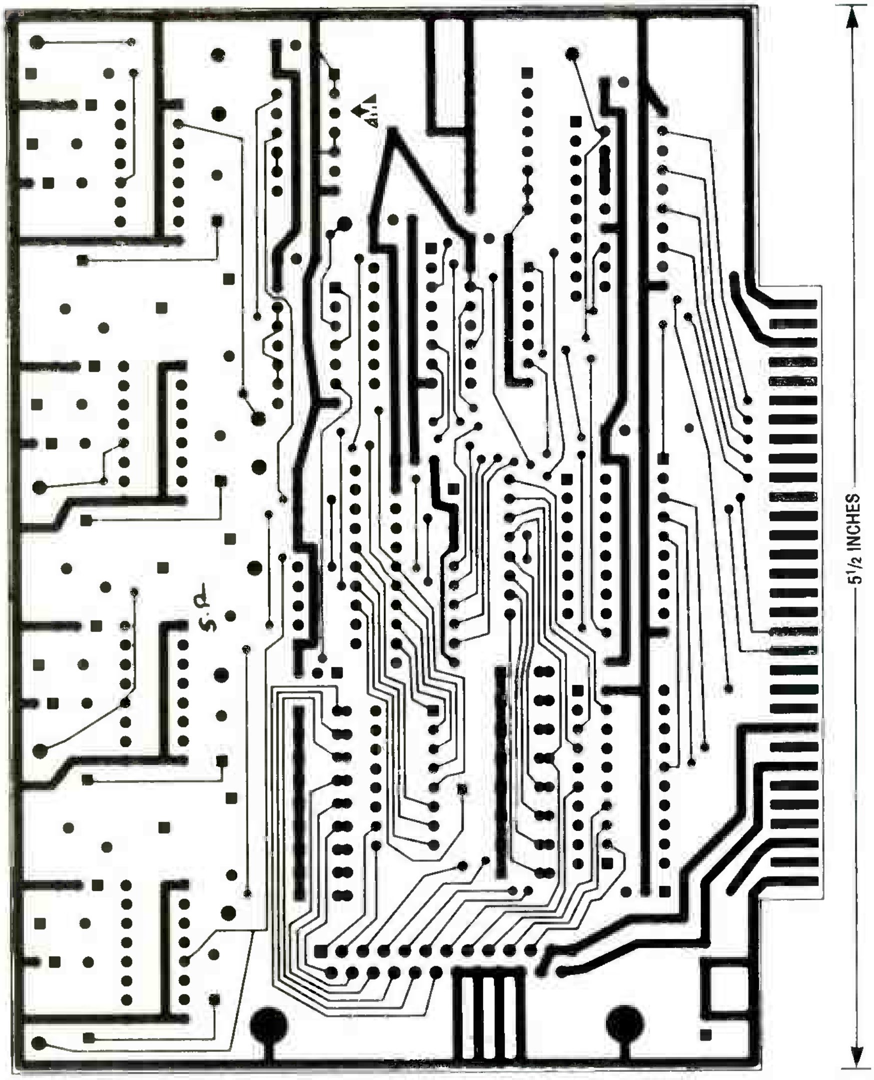


FIG. 8—FRONT-END SCHEMATIC. Each front end contains a data termination block and an address and handshake termination block that use push-on jumpers.



SOLDER SIDE of the I1000 interface board.

FRONT-END PARTS LIST

Resistors

R1—33 ohms, 16-pin DIP resistor
R2—2200 ohms, 10-pin SIP resistor
R3—1000 ohms, 10-pin SIP resistor

Capacitors

C1–C7—0.15 µF, 50 volts, monolythic or polystyrene
C8–C11, C20–C28—1500 pF, 63 volts, polystyrene
C12–C19—220 pF, 100 volts, ceramic disc

Semiconductors

IC1—74LS573D octal latch
IC2—74LS688D 8-bit magnitude comparator
IC3—74LS245D octal transceiver
IC4—74LS02D quad 2-input NOR gate
IC5, IC6—octal buffer
IC7—74LS08D quad 2-input AND gate

Other components

J1—16-pin male header
J2—18-pin male header
J3—Right-angle PC-mount male DB25 connector

Miscellaneous: 17 shorting blocks (for J1 and J2), solder, etc.

Note: The following items are available from TSW Electronics Corp., 2756 N. University Drive, Suite 168, Sunrise, FL 33322 (305) 748-3387:

- **I1000 kit—$65.00**
- **I1000 PC board only—$35.00**
- **I1000, assembled and tested—$77.00**
- **6-foot interface cable (DB-25-6)—$12.95**

Add $3.00 S&H for each order. Send check or money order only.

and handshake buffer restores the original fast rise and fall times of the signals.

Sending a byte

When describing software-related functions, we'll again use BASIC due its wide popularity and we'll assume the following initial conditions:

- The base address of the I1000 is 768 (hex 300).
- The front end of the peripheral has not been selected.
- The address of the peripheral is 4.

Refer to the front-end schematic in Fig. 8 and the following source code:

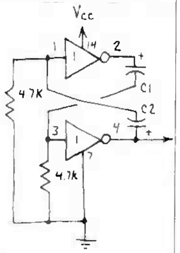
```
10 BAS=768
20 OUT BAS+31,4
30 OUT BAS+2,170
```

Line 10 in that example assigns the address "768" to the variable "bas." Line 20 causes the $\overline{\text{SEND}}$ and $\overline{\text{CREN}}$ pulses at IC6 pins 8 and 9 to go low (refer to the timing diagrams in Figs. 2 and 3). If the shorting blocks have been installed at header J2, then the $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{ADDRESS}}$, $\overline{\text{SEND}}$, and $\overline{\text{CREN}}$ lines are all terminated. Line driver IC6 restores the original wave shape of any signals fed to it. The $\overline{\text{SEND}}$ and $\overline{\text{CREN}}$ pulses exit IC6 at pins 12 and 11. If the shorting blocks have been installed at J1, then the data lines D0–D7 are terminated. Either way, the data is fed to the input of latch IC1; IC3 is inactive at this time. At a time 750 nanoseconds later, the $\overline{\text{WR}}$ pulse enters IC5-a where it is reshaped. It is combined with the cleansed $\overline{\text{CREN}}$ pulse by IC4-d to produce the WR-CREN pulse.

The WR-CREN pulse latches the data (a binary 4) into IC1. The binary 4 appears at the output of IC1 and, subsequently, at the input of IC2, an 8-bit magnitude comparator. The magnitude comparator (IC2) compares the byte fed into it from IC1 with its hardwired address (see the IC2 address-configuration chart contained in Fig. 8). If the two bytes match, pin 19 of IC2 goes low ($\overline{\text{BEN}}$). $\overline{\text{BEN}}$ is then combined with $\overline{\text{SEND}}$ by IC4-b to produce the OUTPUT ENABLE control line signal ($\overline{\text{OE}}$) used by IC3, which transfers all the data to and from the peripheral.

When $\overline{\text{BEN}}$ is high, IC3 is inactive. The BEN line ($\overline{\text{BEN}}$'s compliment) is produced at IC4-c and enables or disables the chip-select section in the peripheral circuitry.

The $\overline{\text{BEN}}$ and BEN lines are the primary lines that determine whether a peripheral on the bus is active or dormant. The direction pin on IC3 (DIR) is controlled by the $\overline{\text{RD}}$ pulse. The $\overline{\text{RD}}$ pulse is high during a write op-



eration, allowing data to flow from the I1000 side of IC3 to the peripheral side of IC3.

Line 20 in the software example activates the peripheral by causing $\overline{\text{BEN}}$ to transition low. Line 30 in the software example will not affect IC1 or IC2. As explained earlier, only an "out" to bas + 31 will activate $\overline{\text{CREN}}$. Line 30 will cause the following sequence of events: $\overline{\text{SEND}}$ will go low. The data (a decimal 170 in this case) will pass through IC3 to the peripheral circuitry. Address information (a decimal 2 in this case) will pass through IC6 to the peripheral circuitry. At a time 750 nanoseconds later, a 500-nanosecond $\overline{\text{WR}}$ pulse will pass through IC5 to the peripheral circuitry. The address is decoded by the chip-select circuit in the peripheral and the $\overline{\text{WR}}$ pulse is then routed to the addressed IC. Any "out" to an address between bas + 0 and bas + 30 will initiate the process commanded by line 30.

Termination

The termination sections are composed of J1, C12–C19, R2, J2, C20–C28, and R3. Those sections provide a termination impedance to ground as well as an R-C time constant. The termination impedance reduces the reflected signal caused by the inductive and resistive properties of the six-foot cable. The R-C time constant slows down the rise and fall times of the signal in the cable, thus reducing crosstalk. As stated earlier, the original transition times are subsequently restored.

Receiving a byte

As we describe how the front end end receives a byte from the I1000 interface, let's assume the following initial conditions:

- The base address of the I1000 is 768 (hex 300).
- The front end of the peripheral has been activated at an earlier time.

Next refer to the following source code:

```
40 A = INP(BAS + 3)
50 A = INP(BAS + 3)
60 PRINT A
```

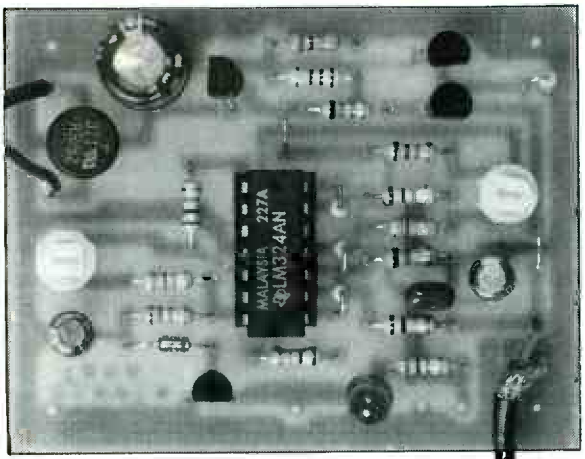
Lines 10–30 are assumed to have been executed previously. Therefore, our theoretical peripheral has already been selected (activated). Line 40 produces a read function as described earlier. The $\overline{\text{SEND}}$ pulse goes low. The address lines (A0–A4) function as they did during the write function. At a time 500 nanoseconds later, a 1-µs $\overline{\text{RD}}$ pulse is received by the front end. It is reshaped by IC5-c and IC5-d. The $\overline{\text{RD}}$ pulse passes through IC7-d to IC3 pin 1. The peripheral side of IC3 becomes an octal input while the I1000 side of IC3 becomes an octal output.

The $\overline{\text{RD}}$ pulse arrives at the read chip-select section of the peripheral circuitry. The $\overline{\text{RD}}$ pulse, in conjunction with the address lines, cause the target IC to place its byte onto the bus. The transmitted byte passes through IC3 to the I1000 where it is latched. A data bus directional delay (DBDD) is provided by IC5-e–IC5-h in combination with IC7-a–IC7-d.

The DBDD provides a delay after the read cycle has finished before returning IC3 to its normal "output" configuration. That prevents IC3's peripheral side from going into its low-impedance state before the IC that was just read is able to de-activate. Line 50 causes the byte latched in the I1000 to be sent to the PC where it is stored under the variable "A." Line 60 prints the value contained in variable "A" on the screen.

As mentioned before, there's no separate front end PC board; each peripheral contains its own front end. Next month you'll see the front-end parts installed on the first peripheral board we'll work on: the T1001. That peripheral contains a 100-MHz frequency counter for digital signals, a period event meter, and a capacitance meter covering 1 picofarad to 10,000 microfarads. Other PC-based test instruments that we will build in future articles, include a logic-IC tester/identifier, and an A/D-D/A peripheral that can also be used as a low-frequency 8-channel digital storage oscilloscope. R-E

electronic THERMOSTAT



This solid-state thermostat can replace those old mechanical units—at a cost of less than twenty dollars!

RODNEY A. KREUTER

THE MEASUREMENT AND CONTROL of temperature is one area in which electronics has had a great impact. From "set back" home thermostats to laboratory controllers with ±0.001-degree accuracy and digital fever thermometers, the use of electronics has all but eliminated mechanical systems.

Many methods are used for measuring and controlling temperature, including the expansion of mercury or alcohol, bimetallic strips, thermistors, silicon sensors, and thermocouples. Each has its advantages and disadvantages.

The author was recently asked to design an inexpensive thermostat to replace some old bi-metallic-type thermostats. The new thermostat had to meet the ±5°C accuracy of the bi-metallic strips, have a −50 to +150°C range, and cost less than twenty dollars. A simple solid-state thermostat was the only solution.

Whether you're trying to keep a fish-tank temperature to within 1°C, maintain working temperature for PC-board etchant, shut down an overheated amplifier, or turn on cooling fans, you'll find that this simple solid-state thermostat will do the job. Note that this project is *only* a controller, so you must supply the heater (or cooler), a suitable relay, and a temperature-measuring device for calibration.

Looking around

Before anyone decides to design and build something, it pays to have a look around to see what's available on the market. First there's the Radio-Shack Thermometer/Controller. Total cost (with switches, etc.) is about twenty eight dollars. The temperature range is −40 to +50°C (−40 to +122°F), and it has a digital readout and temperature memory. So far so good—if the temperature range suits your needs. Maximum measurement speed is once per second. However, the real drawback is that if the temperature limit is exceeded, the output goes high for one minute; during that time period the temperature is not measured!

National Semiconductor has been making a number of temperature sensor/controllers for at least 15 years. The LM3911 (−25 to +85°C) and the LM35 (−55 to +150°C) are two examples. They are easy to work with, but they are more difficult to find and ones with a large temperature range aren't exactly cheap.

Sensors are also made by Linear Technology (the LM134 with a −55 to +125°C range) and Analog Devices (the AD590 with a −55 to +150°C range) as well as dozens of others. The only catch, besides availability, is that they are precision sensors meant to measure as well as control temperature. They are also quite expensive.

Complete controllers are also made by other companies such as Omega, but the cost is about the same as a cheap personal computer. That is due partly to super accuracy and digital temperature readout.