A PC Micro-Development System

Lets you use your IBM PC/compatible computer as a microcontroller development system

The PC that sits on your desk can be used to do a lot more than what you've likely been using it for. If you're into working with microcontrollers, it can be used in conjunction with the build-it-yourself Cyber HC5 Micro-Development System described here to add another dimension to its use. With the system we're about to describe, you can directly program and evaluate your own processor designs without having to purchase expensive equipment. Based on the popular Motorola MC68HC705Cx series of microcontrollers, the Cyber HC5 makes the development process fun, fast and easy to do.

With a PC/Cyber HC5 Micro-Development System, you can develop applications limited only by your imagination and the amount of effort you're willing to devote to a project. Among the diverse things you can do is create the ultimate thermostat for your home, design a new robotics device, even design a new consumer product you might have been thinking about designing for years, to name just a few.

The Preliminaries

If you're new to the microcontroller development process, you may at first find it to be extremely intimidating. Persevere, though, and you'll soon be designing a whole battery of microcontroller applications.

The first thing you should know is the difference between microcontrollers and microprocessors. Basically, microprocessors (the so-called "brains" around which personal computers are built) use external memory. In most cases, they're designed to be used with data and address buses and other peripheral devices. Microcontrollers, on the other hand, usually have internal memory and are designed to directly interface with input and output control lines, rather than having to work through address and data bus-based peripheral interfaces.

Though some people perceive them to be limited in terms of computing ability and the number and variety of operations they can perform, these are the greatest strengths microcontrollers have to offer. By tightly coupling the internal processor directly with sensors and actuators on input and output lines, microcontrollers often provide the most cost-effective and versatile solutions to applications in real world problems.

Since you'll be using the Motorola MC68HC705C8 series of microcontrollers, contact Motorola to obtain a kit of documentation materials. Included will be product documentation and free MC68HC705C8 assembler and Programmer programs. You can obtain this documentation package by writing to: Literature Distribution Center, Motorola Inc., P.O. Box 20912, Phoenix, AZ 8036-0924. Request the Technical Data Book and Applications Guide for the MC68HC705C8. Include a SASE with postage for 1 pound of material.

The documentation you receive will bring you up to speed on the MC68HC705C8. It details some design examples based on the device and gives lots of ideas for implementing your own designs.

About the Circuit

The MC68HC705C8 was designed to "self-program," using a ROM-based "boot-strap" routine. Originally, EPROM-based microcontrollers had to be programmed with a separate, usually expensive, programmer.

Some microcontroller versions had sockets mounted directly on top of the IC to receive a preprogrammed EPROM. The idea was simple. You developed the program code that suited your application and then plugged it onto a special version of the processor. Later, when you debugged your code, the manufacturer would transfer it into a "masked" ROM. For low-volume runs, this process was time-consuming and expensive and, thus, not very practical.

By developing a microcontroller that could program itself (transfer ROM code directly into its own internal EPROM), Motorola now allows you to program one device or thousands in a very cost-effective manner. The "bootstrap" ROM routine, located inside the MC68HC05C8 package, handles details for programming the internal EPROM. All you need to facilitate the process is the hardware to interface the MC68HC705C8 with your computer, which is the task of the Cyber HC5.

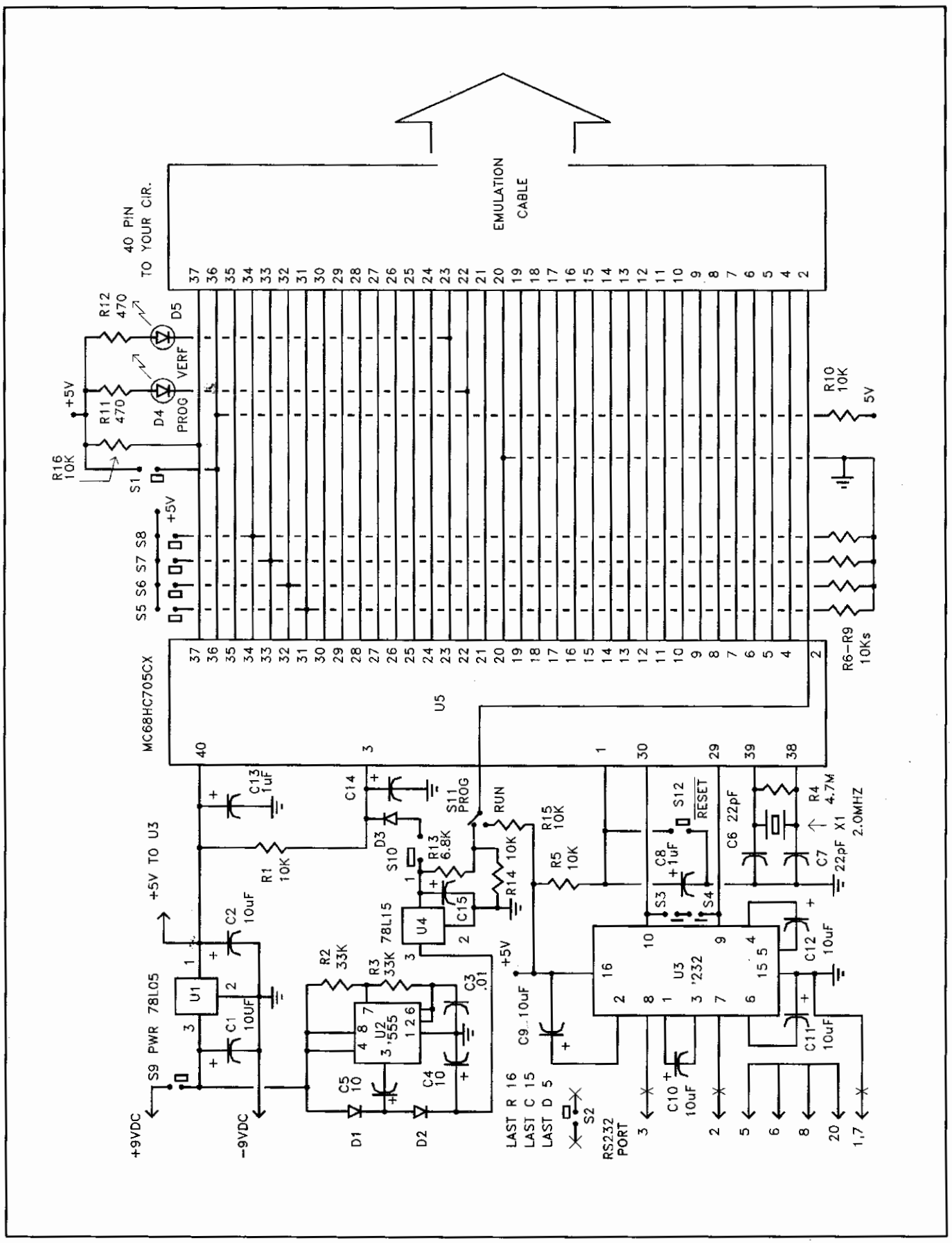


Fig. 1. Complete schematic diagram of circuitry used in the Cyber HC5 Micro Development System.

PARTS LIST

Semiconductors
U1—78l05 fixed +5-volt regulator
U2—NE555 oscillator/timer
U3—MAX-232 RS-232 interface
U4—78l15 fixed +15-volt regulator
U5—MC68HC705C8E EPROM version
D1,D2,D3—1N4148 signal diode
D4—Red light-emitting diode
D5—Green light-emitting diode
Capacitors
C1,C2,C4,C5,C9,C10,C11,C12,C14, C15—10-μF, 16-volt tantalum
C3—0.01-μF, 100-volt Mylar
C6,C7—22-pF, 100-volt ceramic
C8,C13—1-μF, 16-volt tantalum
Resistors (¼-watt, 5% tolerance)
R1,R5,R10,R14,R15,R16—10,000 ohms
R2,R3—33,000 ohms
R4—4,700,000 ohms
R11,R12—470 ohms
R13—6,800 ohms
R6 thru R9—Three elements inside 10,000-ohm SIP network
Miscellaneous
PS1—9-volt, 100-mA dc plug-in power supply
X1—2.0-MHz crystal
S1 thru S8—Eight-position DIP switch
S9 thru S12—Dpdt slide switch
SK1—40-pin IC socket
ZIF1—40-pin ZIF (zero-insertion force) socket
CONN1—DB-25 male connector
Printed-circuit board (see text); suitable enclosure (see text); three-conductor shielded cable; 25-conductor ribbon cable and IDC connectors for emulation cable; solder; etc.

Note: The following items are available from U.S. Cyberlab, Inc., Rte. 2 Box 284, Cyber Rd., West Fork, AR 72774; tel.: 501-839-8293 or 1-800-232-9865: Ready-to-wire pc board, $19.95; complete kit of all parts for Cyber HC5, including pre-cut case and front panel, $89.95. Also available is complete Cyber Lab Breadboarding System for all microcontrollers/microprocessors, $89.95. Add $4.95 for P&H per order. Arkansas residents, please add 5% sales tax. MasterCard and Visa welcome.

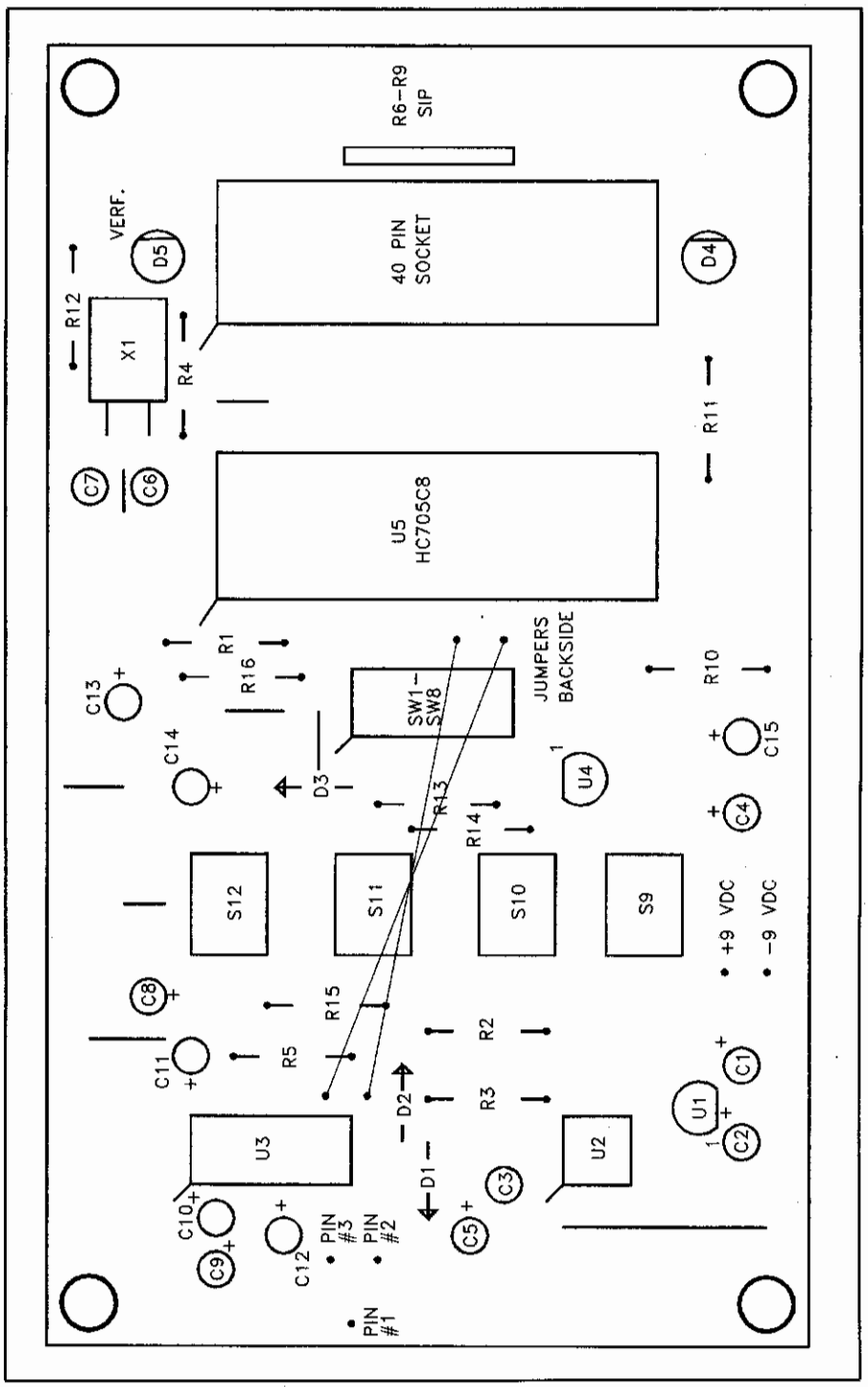


Fig. 2. Actual-size etching-and-drilling guide for the Cyber HC5's pc board.

Shown in Fig. 1 is the complete schematic diagram of the circuitry used in the Cyber HC5. Power is supplied to the circuit from a plug-in external 9-volt dc module. The +9 volts is applied to the circuit through POWER switch *S9* and is regulated to +5 volts by regulator *U1*. Unregulated +9 volts is also applied to oscillator *U2*.

Regulated +5 volts from *U1* supplies *U5* and RS-232 interface *U3*. Unregulated +9 volts powers voltage doubler *U2*. The output of *U3* is held to a constant +15 volts by regulator *U4*, which supplies V_{pp} to *U5*.

Chip *U3* and charge-pump capacitors *C9* through *C12* form a self-contained RS-232-to-digital interface. Crystal *X1*, a 2-MHz device, controls *U5*'s internal clock circuit and permits communication with the host computer at 4,800 baud.

With an MC68HC705C8 plugged in the *U5* ZIF socket and power applied to the circuit, a voltage across *R5* rapidly brings high *C8* and forces a hardware reset at pin 1 of *U5*. Setting *S11* to PROG, causes *U5* to sample the input at its pin 2 and "sees" a voltage

greater than V_{dd} (+5 volts). This tells the microcontroller to go into self-programming mode.

In self-programming mode, *U5* executes the internal "bootstrap" communication routine that subsequently reads the input at Port D on pins 31 through 34 of *U5*. These pins are controlled by the settings of *S5* through *S8*.

"Load program via SCI to RAM and execute mode," causes *U5* to begin downloading the required program and data from the host computer through *U3*. Later, when prompted by the control program running on your host computer, you set *S10* to PROG to place +15 volts on pin 3 to program the internal EPROM. If it's necessary to reset the above process, set *S12* to RESET to short pin 1 to ground and force *U5* to reset.

Transfer of data to and from *U5* can be visually monitored via light-emitting diodes *D4* and *D5*. These LEDs light to provide visual verification of the PROGRAM and VERIFY functions.

As you can see, almost every pin on the MC68HC705C8 plugged into the *U5* socket is connected to a 40-pin IC socket shown to the right. This socket allows you to connect a ribbon cable from the Cyber HC5 to your prototype hardware, eliminating the need to remove *U5* from the Cyber HC5 every time you wish to test it. Simply move *S15* to RUN and then use *S12* to reset the device. It automatically begins executing the new program internally and operates as though it's installed in your prototype unit.

Construction

Though you can build the Cyber HC5 on perforated board that has holes on 0.1" centers, using suitable Wire Wrap or soldering hardware, you'll be much better off if you use a printed-circuit board. You can fabricate your own pc board using the actual-size artwork shown in Fig. 2. Alternatively, you can purchase a ready-to-wire board from the source given in the Note at the end of the Parts List.

Wire the pc board exactly as shown in Fig. 3. Begin populating it by mounting and soldering into place the DIP IC sockets, followed by the resistors, capacitors and jumper wires. If you, wish, you can plug the ICs into their respective sockets now or wait until after you've conducted your preliminary voltage checks and are certain that all wiring is correct.

Note that the SIP that contains *R6* through *R9* can be an eight- or nine-pin type. During mounting, simply fold the pins that aren't used under the SIP on the top side of the pc card.

When you install *S9* through *S12*, be sure to space them vertically and horizontally across the board. Use the front-panel cutouts in Fig. 4 to line up the switches for a neat appearance.

Mount and solder into place 78L05 regulator *U1*. Then use a dc voltmeter or the dc-volts function of a digital multimeter to check the output of the powered 9-volt dc supply module. Mark the polarity of each lead. Unplug the supply from the ac line and plug the conductors of its output cable into the polarity-correct holes in the pc board and solder them into place.

Make two or more same-size photocopies of the front panel artwork shown in Fig. 4 and trim them to size. Then use rubber cement or contact spray (Scotch No. 77) to adhere one copy to the front panel of the enclosure. Use this to transfer the hole dimensions to the front panel of the enclosure.

Use a hot knife to cut the switch and IC holes and a slot at the bottom of the enclosure to accommodate the 40-con-

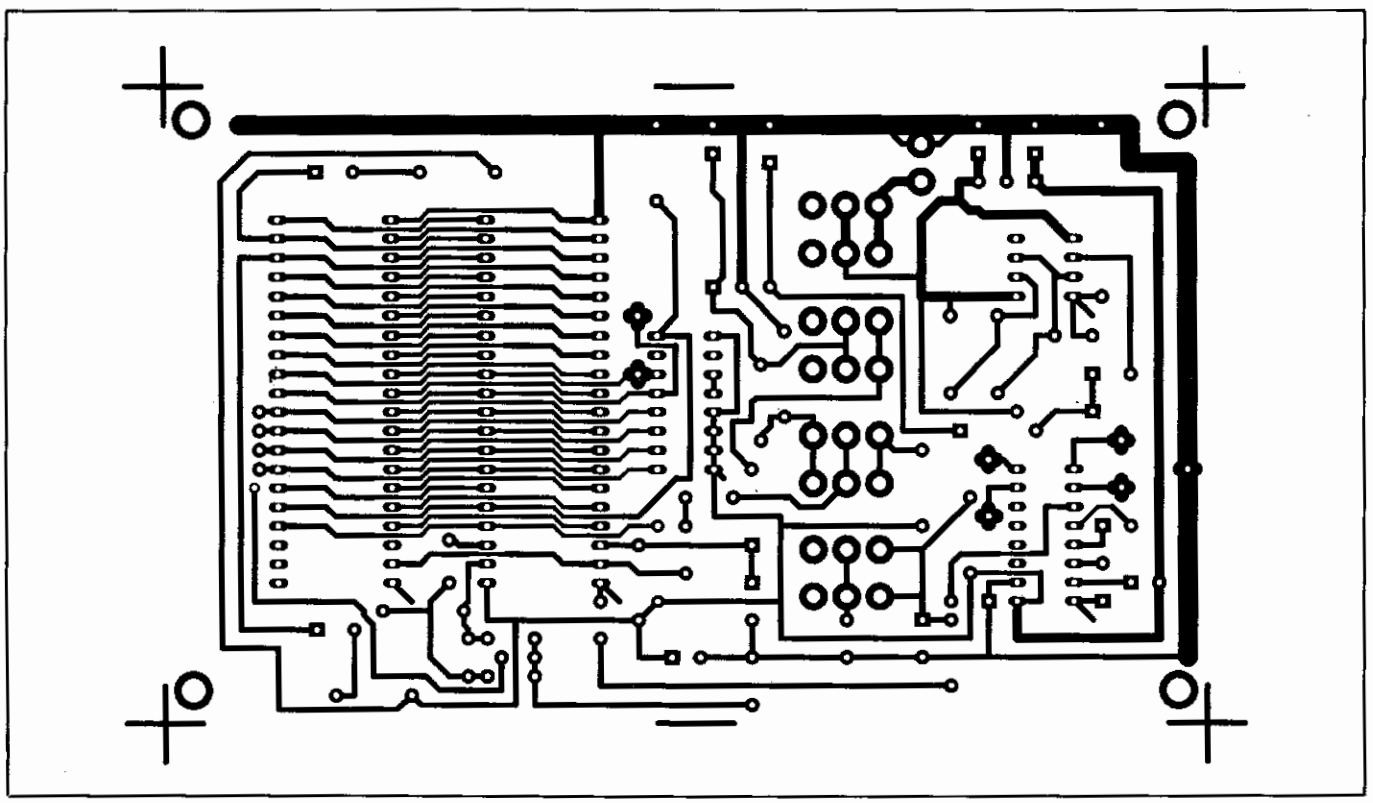


Fig. 3. Wiring guide for the pc board.

ductor ribbon cable option. (A hot-knife is a small razor-sharp knife fitted to the end of a soldering iron and is used to cut precise holes in plastic. If you don't already have one, you can obtain a hot knife from a hardware store. It's a good investment. You'll use it whenever you build a project that requires a home-machined plastic enclosure.)

When using a hot knife, work very carefully. If you haven't used a hot knife before, start by making the holes a little under-size until you're comfortable with the cutting process. (A precut enclosure can be obtained from the source given in the Note at the end of the Parts List if you wish to avoid having to machine your own.) You can trim to final dimensions later. When you're finished using it, always remember to thoroughly clean the tip of your hot knife, allow the knife to cool and stow it safely for the next time you need it.

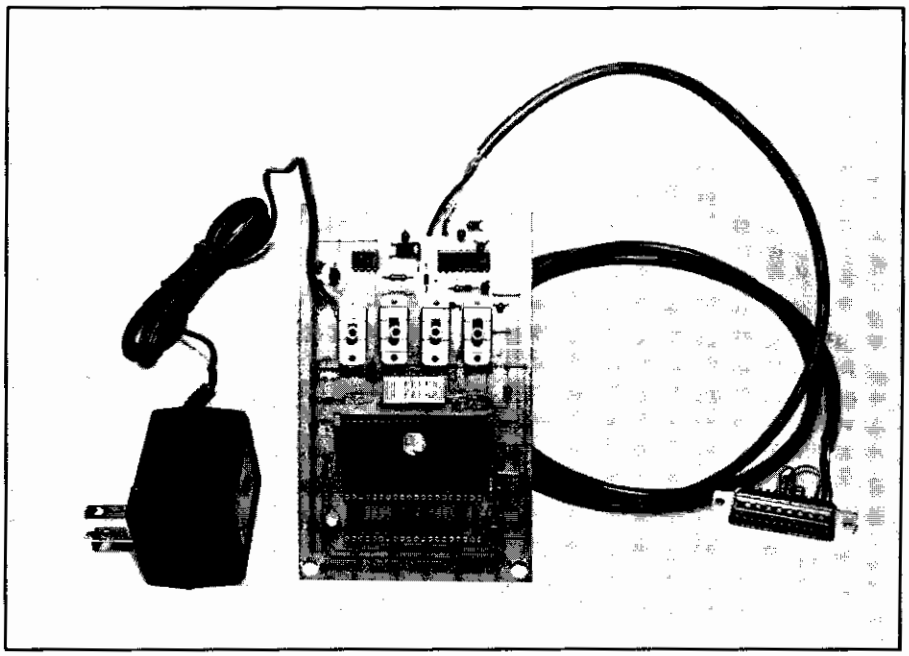
Test fit everything in the enclosure. When you're satisfied with the hole sizes and spacing, peel away the first photocopy of the artwork from the front panel of the enclosure. Trim the second photocopy, (or the membrane applique for the front panel provided with the kit, if you purchased it) for a perfect fit on the enclosure's front panel. Then use spray adhesive to secure the artwork in place.

It's sometimes useful to run a No. 2 pencil around the inside of the front panel holes, where the edge of the front panel applique meets the black plastic. This masks the cut marks and blends the front panel into the enclosure for a factory-finished appearance.

Now assemble the RS-232 interface cable and connector. Referring back to Fig. 1, note that you must connect together pins 5, 6, 8 and 20 using No. 22 hook-up wire. Connect the RS-232 cable to pins 1, 2, 3 and 7, as detailed. Depending upon your particular host computer serial interface configuration, you might have to swap pin 2 and 3, as in a "null" cable arrangement.

When you finish wiring the connector, run the serial cable through the hole at the top of the Cyber HC5 case and solder its conductors to the pc board, as shown in Fig. 3.

As an option, you may want to install a 40-conductor ribbon cable between the 40-pin IC socket and your project. In this case, you should use a 40-pin IDC header at both ends of the emulation cable. Using this option allows you to plug the Cyber HC5 cable into the 40-pin processor socket on your project. When you finish programming and debugging your software, move the programmed MC68HC705C8 from the *U5* socket on the Cyber HC5 to your project.



Assembled Cyber HC5 is ready to be installed in its enclosure.

Designing a Project

Before you learn how to use the Cyber HC5, you should become familiar with the design process you'll be using. The procedure is as follows:

Step 1. No matter how simple or sophisticated a project, begin by conceptualizing it in your mind. Don't put too many limitations on your first thoughts. You'll use Step 2 to filter out what can and can't reasonably be accomplished. Sketch out your ideas and concepts on paper, deciding on the basic hardware you need in addition to the MC68HC705C8. Check the Motorola documentation examples to get a "feel" for how your software will be structured. Be creative, and spend as much time as you can in thinking through each of your project's features and where they'll lead.

Decide on which features are musts and which are "bells and whistles". Include the latter only if the hardware makes them feasible and you can afford to include them. If this is your first time around with this thinking process, don't worry about making mistakes. Every product you've ever seen and every project ever designed goes through changes between conceptualizing them and bringing them to fruition.

Step 2. Now turn on your design "filters." Draw a schematic diagram of your project, using the MC68HC705C8 documentation. Use the design examples provided to determine what to do with pins on the processor that aren't used. If you don't initially understand how a particular processor function or peripheral works, the documentation should clear the air.

With schematic finished, build the project. (The Cyber Lab Breadboarding System mentioned in the Note at the end of the Parts List is an ideal testbed to use during this stage. With it, you can quickly breadboard your circuit and interface it directly with the Cyber HC5.)

Take your time building the prototype. Keeping in mind that it may require extensive modification as it evolves, it's important to keep your work modular so that sections of the circuit can be changed and enhanced without affecting the entire system.

Step 3. If you're familiar with BASIC, Fortran, COBOL or other programming language, you have a good head-start on things. Programming experi-

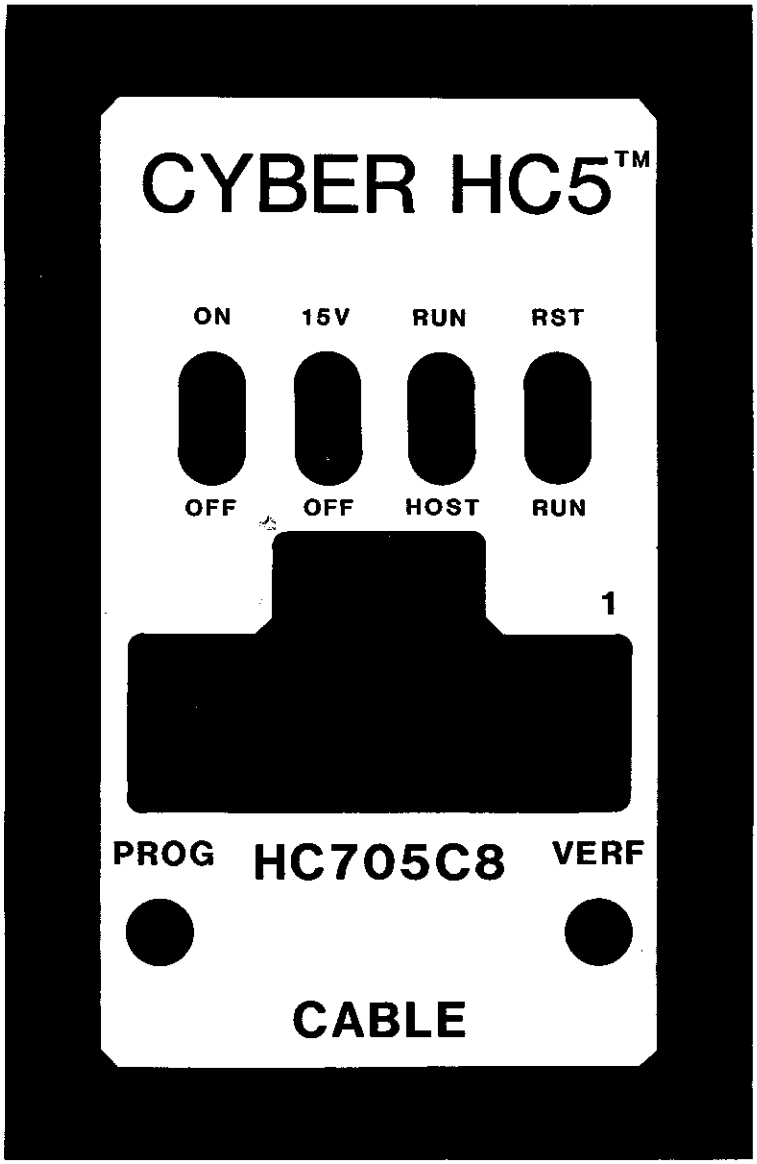


Fig. 4. Actual-size artwork for the front panel of the Cyber HC5 enclosure.

ence in any of these high-level languages is helpful. Machine language, machine code, assembler and assembly language are all names commonly applied to what you're about to become involved with. Don't rush things. Use the Motorola documentation to carefully work your way through the example listings, and use the exact listings from the book if you feel they're applicable in your particular project.

Assembly language gets its "tough" reputation because of the amount of detail involved. By taking your time to work through each instruction (called the "single-stepping process"), you'll catch most of the errors you'll make before you "burn" a defective program into EPROM. *Never* assume anything when working in Assembler! If a particular detail isn't completely clear to you, don't assume the processor can sort it out.

I often write a very simple input/output routine for my new projects. Using the Cyber HC5 it's an easy matter to re-program my MC68HC705C8 each time I make a substantial change to the control program. Keeping my programs modular, (dedicating individual sections, or blocks of memory, to particular functions) makes it easier for me to debug my programs.

If you know how to program in C, a C compiler is available for the MC68HC705C8. The C language is almost ideally suited to this application. I highly recommend making the move to C if you haven't already done so.

Motorola provides all new users of the MC68HC705C8 a freeware version of its '05 Assembler. Although it isn't the most powerful assembler you can get, it's free. If you don't receive a copy of the assembler with your documentation package, call Motorola's Freeware BBS (512-891-3733) and download it. You must set your modem to 300, 1,200 or 2,400 baud. Download using KERMIT, Xmodem or any other protocol Freeware BBS supports. Details for operating the assembler and programmer programs are included.

Step 4. Begin programming your first device by plugging in the Cyber HC5 and setting its POWER switch to OFF. Then, connect the DB-25 connector to the serial communications port on your host computer.

The next step is very important and *must* be done each time you use PROG7. Use the DOS MODE command to configure your serial port. It may be necessary to copy MODE from your DOS disk or directory to the disk or directory that contains your assembler and programmer programs. Key in MODE COM1:4800,N,8,1 and press ENTER to open your serial port at the proper baud rate for the PROG7 program. Make sure your serial card is configured as COM1, or change the card or command.

Next, run the PROG7 freeware program. Select <C> from the main menu to check the Cyber HC5 connection to your computer *without* the MC68HC705C8 plugged into the *U5* socket on the Cyber HC5. Turn on power to the Cyber HC5 and set *S1* in the eight-position DIP switch to open and leave it there from now on. (Switch *S1* will always be open when you use the Cyber HC5 for programming the MC68HC705C8.) Switch *S2* isn't connected to anything and isn't used in the Cyber HC5. Close switches *S3* and *S4*. This shorts or "loops-back" serial port data from the computer to the host. (Make sure the MC68HC705C8 isn't plugged into the *U5* socket during this test.)

Select the loop-back test from the PROG7 menu and press ENTER. Whatever you now type on your computer's keyboard should be "echoed" on the screen of your video monitor.

Table 1. MC68HC7O5C8 Operating Modes Using Cyber HC5

S5	S6	S7	S8	Mode
On	X	X	X	Execute program in RAM at $0051
Off	Off	On	Off	Load program from host to RAM and execute
Off	On	On	Off	Dump EPROM contents to host
Off	On	On	On	Secure EPROM and Dump to host

If echoing doesn't occur, you might have to transpose the connections to pins 2 and 3 on your DB-25 connector.

With the Cyber HC5 echoing all characters, you can begin your programming session. The following is the procedure for programming the EPROM in the MC68HC705C8.

(1) Set the positions of the DIP switch as follows: *S1* to open; *S2* to open; *S3* to open; *S4* to open; *S5* to open; *S6* to open; *S7* to closed; *S8* to open. This configures *U5* to accept data from the host computer and automatically program itself.

(2) Set the POWER switch to OFF.

(3) Set the RESET switch to RST.

(4) Set the V_{pp} switch to OFF.

(5) Set the HOST switch to HOST.

(6) Plug the MC68HC705C8 into the *U5* ZIF socket. Make certain pin 1 is in the correct position.

(7) Set the POWER switch to ON.

(8) Set the RESET switch to RUN.

(9) Use the PROG7 program to select the "blink" check. (PROG7 prompts use different switch numbers than Cyber HC5.)

(10) The two LEDs should begin to blink and then stop after about 16 seconds to indicate that *U5* is ready to be programmed.

(11) Select <L> to load your assembled program and then <F> for file. Be sure to select <F> to flush the <O> ROM buffer first. Then select <R> to read your file to be programmed to <O> ROM. The assembler will generate a file called "myprog.S19," where *myprog* is the name you gave to your program. The .S19 extension means the program contains the "S" record file data that's a Motorola standard used to exchange binary data between devices. PROG7 uses this data to program the MC68HC705C8.

(12) Select <P> to program *U5* and <O> to indicate ROM. Select to <B> blank-check *U5* (to make sure it has been erased prior to programming it. With a blank MC68HC705C8 plugged into the *U5* socket, select <P> to program and <B> to "blast" the device. Carefully follow PROG7's menu. When it asks you to apply the +15 volts, move the V_{pp} switch to the +15 volts position. When programming is done, set the V_{pp} switch back to OFF.

(13) The program automatically verifies the contents of *U5* and compares it with the myprog.S19 data, which should match. If it doesn't, repeat the above process. When you're done, power down.

Step 5. With a programmed MC68HC-705C8, you can decide whether or not you want to physically move it from the Cyber HC5 to your hardware project or use the 40-conductor emulation cable on the Cyber HC5 to test *U5* in-place. If you want to use the Cyber HC5 in the emulation mode, connect the 40-conductor cable and header to the 40-pin socket on your hardware project. (Be sure to connect and reconnect this cable each time you program. In some cases, the programming process could interact with your hardware to affect the process.)

Next, set the RESET switch on the Cyber HC5 to RST and the HOST switch to RUN. Apply power to your hardware and then the Cyber HC5. When you're ready to view execution of your program, set the RESET switch to RUN and watch as *U5* automatically begins executing your program.

Step 6. Debugging your project will often be the most time-consuming part of the design process. Testing, testing and more testing is the order of the day. Each time you find a "bug," repeat Steps 3, 4 and 5. If you keep at it, you'll get it.

At this point, you should be aware of some other Cyber HC5 operating modes. Table 1 lists the modes the MC68HC705C8 can be forced to assume by setting the positions of *S5* through *S8* accordingly. "Load program from host to RAM and execute" is the mode used to program *U5*. PROG7 works by downloading a byte-transfer routine to *U5* and executing it.

Other modes exist for the MC68HC-705C8, but aren't used with the Cyber HC5. I think you'll find you can do about anything you want using the PROG7 routine. Just be sure to use the MODE COM1:4800,N,8,1 command line before running PROG7; otherwise, you'll chase your tail for hours.

Although you may find it a little difficult to program at first, persevere because the HC705C8 provides you with a powerful microcontroller for your designs. This inexpensive chip lends itself well to high-volume applications. Contact Motorola about mask-ROM parts and pricing. ■

Nick Goss