



## FEATURE ARTICLE

Edward Cheung

# A Winning Combination

## PIC Internet Connectivity

Edward's winnings in the Internet PIC2000 design contest turned out to be quite a treat for him, but now we're in for a treat of our own as he explains some of the tricks that went into designing these award-winning Internet projects.



When I entered the *Circuit Cellar* PIC2000 Contest sponsored by Microchip,

I saw many possibilities. First, I developed the PIC Web Server, which needed a good application in order to come to life. I developed five Internet applications that could be hosted on the PIC. Four of these projects won—including the top three prizes.

Home automation has been my hobby for a long time. More than 10 years ago, I started with a collection of lamp and appliance control modules from X-10. I gradually added components to the system, including a central desktop computer running custom home control software that communicates with a network of PIC microcontrollers. The network uses the RS-485 standard, which is similar to HCS-II. It interconnects network nodes that do a variety of real-world input and output functions.

I can control the system from any phone, but I wanted to control it from the Internet. Ideally, I would add the Internet/Ethernet interface at a node on the RS-485 network, freeing the desktop computer for other tasks. The thought of a PIC communicating via the local Ethernet network was appealing because it could be used in other applications too.

In July 1999, *Circuit Cellar Online* published an article describing a microcontroller web server. [1] It was based on an Atmel processor interfaced to a standard NE2000-compatible PC Network Interface Card (NIC). The idea was brilliant: take a commonly available Ethernet interface card, use the microprocessor to emulate its bus interface, and you have a microcontroller Ethernet node. That was the proof I needed to create a PIC version.

My finished product is a web server that is roughly the size of two video-cassettes. All it needs is power and a 10BaseT connection to provide any user on the local network with real-world interface to his web browser.

The first step in constructing a web server on a microcontroller involves knowing the protocols used on the Ethernet network. I wanted to build a web server from only web resources. It probably stretched the process, but it was more fun to learn my way from the tidbits scattered about the 'Net. In the end, I obtained an understanding of the entire process—from the top level where a user clicks in a URL box of a browser, down to the level of the voltage sent on the physical Ethernet wire.

### WEB SERVER HARDWARE

The NIC uses the PC-ISA standard for its bus interface. The PIC emulates this bus so you can communicate and control it. All 16 bits of the data bus

Table 1—Take a look at the Ethernet packet structure. Nodes on the local network can decode the recipient and sender's hardware address.

Name	Length	Comment
Preamble	62 bits	Prepended by NIC
Start of frame (SFD)	2 bits	Prepended by NIC
Destination address	6 bytes	Ethernet address
Source address	6 bytes	Ethernet address
Type	2 bytes	Length data for IEEE 802.3
Data	46–1500 bytes	Least significant bit and byte first
Checksum	4 bytes	Calculated and appended by NIC

and 5 bits of the address bus are connected to the PIC. The remaining bits of the address bus are strapped either high or low to presume the base address of the NIC is at 0x300. The number of address lines allows the addressing of 32 registers in the NIC. Depending on the initial setup of the NIC, it may have to be plugged into a computer to disable plug-and-play and to set the base address to the desired value. In addition to these lines, the minimum set of ISA bus control lines is connected to the PIC (see Figure 1).

I chose the P16F877 as the micro-processor because of its flash program memory and the availability of the In-Circuit Debugger (ICD). The 8 KB of program space was just enough to fit the web server and project application code. A Maxim chip provides the voltage level translation needed for RS-232 (for the PIC Web Cam project), and two of the PIC's I/O lines are

Name	Length	Comment
Type	1 byte	Eight for echo request, zero for echo reply
Code	1 byte	Always zero
Checksum	2 bytes	One's complement type
Identifier	2 bytes	Used for matching messages
Sequence number	2 bytes	Used for matching messages
Data/payload	Variable	Data to be echoed

Table 2—The TCP packet is sent within an Ethernet packet.

brought to an RJ-11 connector (for the Chart Recorder project and X-10 interface). There are no additional RAM or ROM chips, which meets my goal of a minimal circuit.

## ETHERNET, FROM THE BOTTOM

At the lowest level, Ethernet is a party line shared by all the computers on the local network. The bits are sent Manchester encoded, which means the logical ones and zeros are not sent as voltage levels (as in RS-485), but as rising or falling edges in

the middle of the bit time. In this manner, both data and clocking information is sent to the receiver.

By comparing the sent and received data, the NIC can quickly detect a collision and release the line. The senders then wait a random amount of time. After that, they retry and await confirmation that the data was sent without contention. The packets are separated by a pause cueing the other nodes to try sending their data. The first data sent is the least significant bit of the least significant byte.

The bit time is 100 ns for 10BaseT and its variants, such as 10Base2.

The structure of the packets is shown in Table 1. The NIC used is compatible with the common NE2000. It has a 16-KB circular buffer to hold the Ethernet data for the host processor. Of the listed data, the preamble and the SFD are not routed into this circular buffer. This data is sent to bit synchronize other NICs on the network. The destination and source addresses at the start of the packet allow nodes to sort the routing of the packet. This 6-byte address is unique for every NIC made. If the most significant bit of the physical address is one, this message is a broadcast and should be processed by all nodes.

The two bytes in the Type field that follow indicate the kind of packet. All data up to

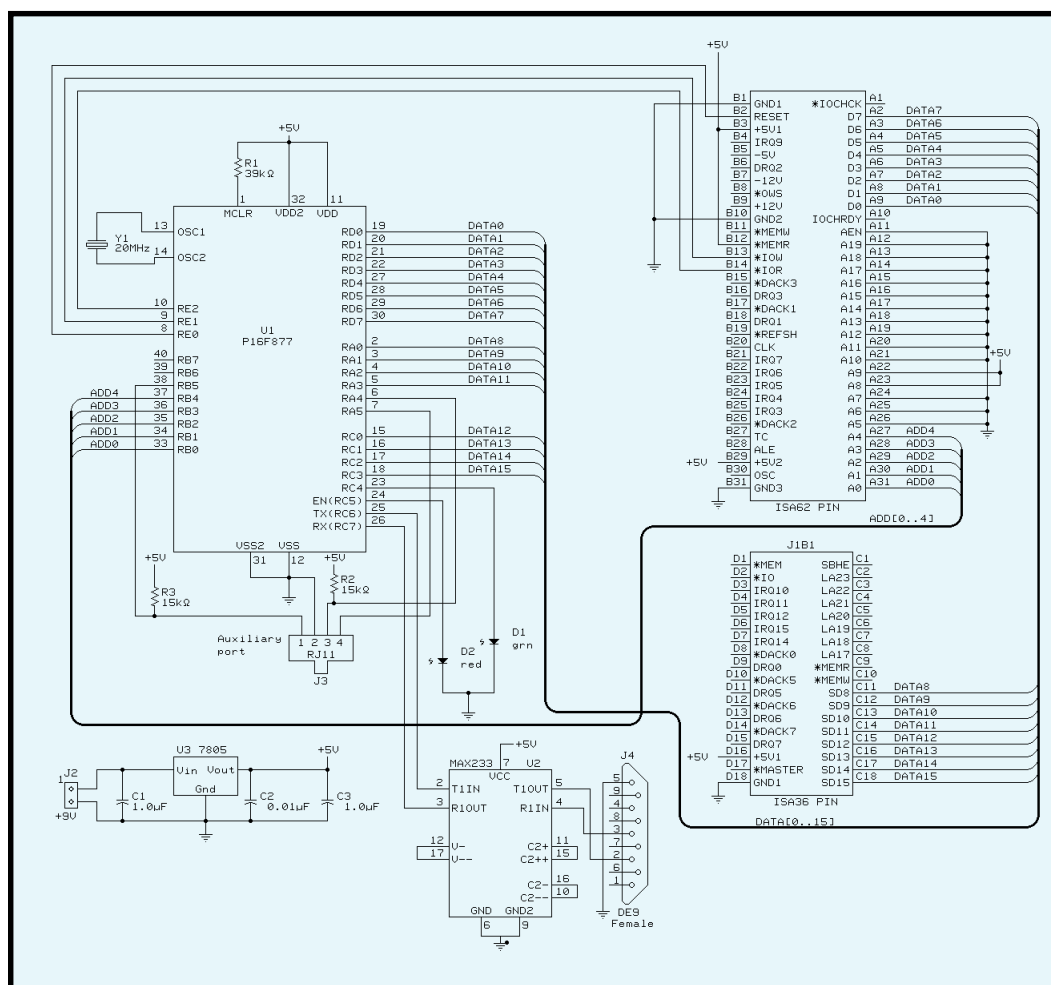


Figure 1—Here's the PIC Web Server. The PIC controls the Ethernet board by emulating the PC-ISA bus interface. All 16 data, five address, and a minimal set of control lines are implemented. U2 provides the level shifting needed for RS-232.

this point (not including preamble and SFD) is referred to as the Medium Access Control (MAC) header. The actual data follows this. Packets shorter than 46 bytes need to be padded to fill the difference. Data packets longer than 1500 bytes do not conform to the standard. In practice, the minimum data packet is larger (60 bytes). Finally, the CRC checksum is calculated by the NIC and appended to the packet. National Semiconductor offers a good discussion of the lowest level of Ethernet and the registers in the NE2000 board. [2]

This layer allows you to send packets from one computer to another using various protocols. One common protocol is Internet Protocol (IP), denoted by "0x0800" in the type field of the MAC header (see Table 2). Figure 2 shows the IP packet structure. [3]

Network nodes that receive this packet can decode its total length and the IP addresses of the sender and receiver among other parameters. Suppose a node on the network needs to send an IP packet to another machine. One problem is that you may not have the MAC (hardware) address of the destination machine; you may only have its 4-byte IP address. This is resolved by using the Address Resolution Protocol (ARP). An ARP request on the network essentially asks: "Which computer on the network has the following IP address? Please respond with your physical address." ARP messages are embedded in Ethernet packets (see Figure 3). [4]

The sender of the ARP request uses a broadcast destination address and "0x0806" in the type field of the MAC header. If there is a network node with a matching IP Address, it will respond with an ARP reply filled in with the appropriate information. Often the reply data is cached to keep the number of ARP messages low. To view the ARP cache for your Wintel box, type "arp -a" at the DOS prompt.

Now that you can send IP messages from machine to machine, you need something to send. One type of IP message is Internet Control Message Protocol (ICMP), often used for ping applications. Most machines that have a TCP/IP stack running will re-

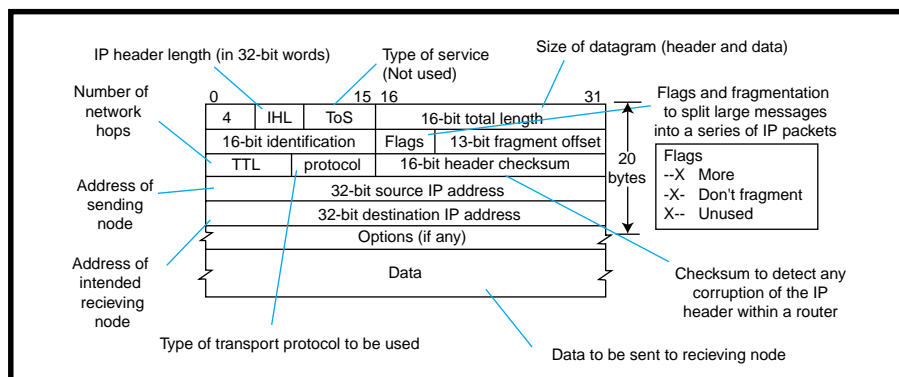


Figure 2—Sent within the data field of an Ethernet packet, the IP packet allows the routing of data to an IP stack running on a local network computer.

spond to these requests by echoing the data. Ping often is used to verify connectivity to other computers on the network.

Another type of packet that can be sent inside an IP packet is UDP. UDP is usually used with streaming data, and is unreliable—the sender does not know if all the data is received properly. (However, this is adequate for certain types of data.) Packets such as UDP are "connectionless", meaning a specific connection is not established between the sender and receiver.

Finally, the most commonly known IP protocol is TCP, which is used with web transfers, e-mail, and so on. C programmers will recognize the TCP/IP protocol as being an implementation of the sockets library, where you need to open a connection to another machine before sending data. There also is a handshaking mechanism where the receiver acknowledges every packet sent. Missing packets are then retransmitted. This mechanism ensures that data is sent in a guaranteed fashion.

Note that TCP/IP packets are not only sent to particular hardware and IP

addresses, but also to a port address. Commonly known applications have standard port numbers. For example, web servers tend to listen for connections at Port 80. Telnet servers use Port 23. In this manner, packets can be routed to individual applications running on a particular computer. As you see, these protocols are layered on top of the previous ones and placed into the Data field of the lower-level packets. Check the references and resources included to learn more about various fields in the packet structure.

## THE NE2000 INTERFACE

I could not find any documents describing the actual NE2000 standard on the 'Net. The most useful resource is National Semiconductor's data-sheet on the DP83905 AT/LANTIC Ethernet chip. [2] The NE2000 has registers that allow you to set up its configuration, such as its MAC address for filtering Ethernet messages and 16-KB circular buffer. This important resource is where outgoing data is queued and incoming data is held until the PIC retrieves it.

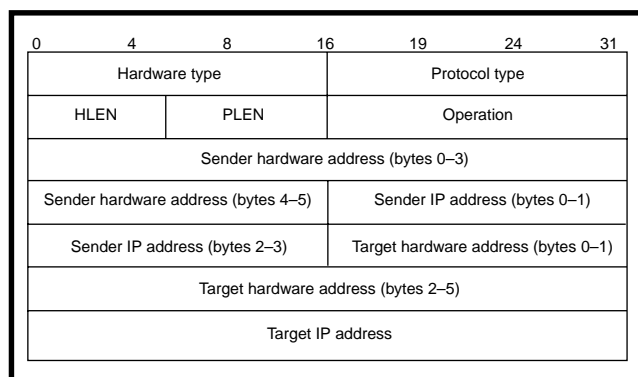


Figure 3—The ARP packet is sent within an Ethernet packet to exchange hardware address information.

After the appropriate pointers are set up, data can be read from or written to the circular buffer through one particular register. The pointers are then autoincremented for the next location. Data exchange is 16-bits wide, and all data is simply the contents of the structures described here. In addition to the conventional use of this buffer, the pointers can be configured to leave a portion unused for general storage.

## GENERIC SERVER SOFTWARE

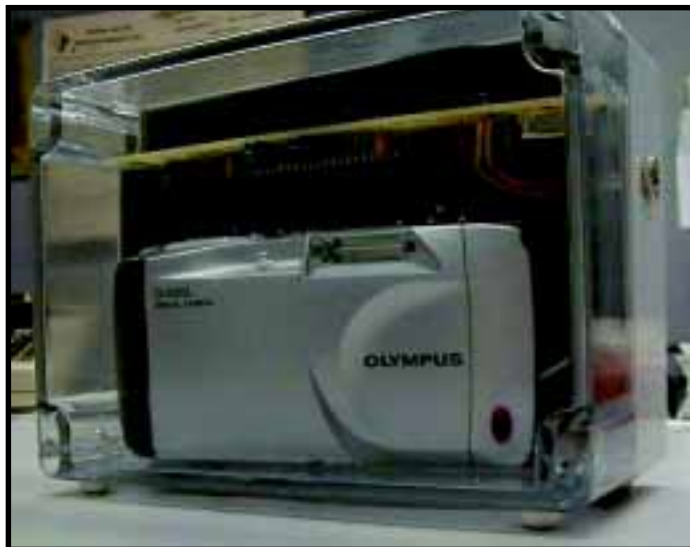
The code for the generic web server occupies about half of the 16F877's program space, leaving room for the application code. It is written in C and compiled using CCS's PCM compiler.

The RAM inside the PIC is not used to store Ethernet data. Rather, data that needs to be sent is built up inside the Ethernet card by moving around the write pointer as needed according to the packet structure. If data that takes into account a series of bytes (such as TCP checksums [5]) needs to be prepared, the data is read back one byte at a time while processing. Hence, the NE2000 buffer doubles as PIC memory.

Web clients and servers have a protocol of their own that dictates requests and replies follow an orderly exchange. The PIC Web Server complies with the HTTP 1.0 protocol. [6] This data, as well as HTML files, are sent from server to client in textual form. One programmer-friendly feature incorporated into the web server code is the way text strings are sent to the client. This form of data can be entered into the source code as follows:

```
printf(...,"HTTP/1.0 200\r\n");
```

C programmers will recognize this as a way to output string data. CCS's compiler allows a variant where the name of a subroutine can be substituted for the ellipsis. This causes that function to be called repeatedly for



**Photo 1**—The PIC Web Cam won grand prize in the Circuit Cellar PIC2000 design contest sponsored by Microchip.

every character in the resultant string. By substituting `ne_print()` (transfers data one byte at a time), whole HTML files can be entered by copying and pasting into the `printf` statement.

Building web pages can be easy. Using a text or a WYSIWYG web page editor, format the web page. Then, copy the raw HTML source from the editor into specially prepared `printf` statements. Finally, compile and run your code. This method doesn't use much program memory space.

Network nodes communicate with the PIC Web Server by first issuing an ARP request with the PIC's assigned IP address. The PIC then responds with its MAC address, allowing future packets to be properly addressed. The TCP/IP stack on the client machine opens a socket to the PIC by issuing a TCP/IP message with the appropriate flags set. Next, the PIC accepts the socket connection and confirms the action. When the socket is open, the web browser submits a GET request containing the full URL as typed by the user in the browser Location box.

This information is then parsed by the PIC Web Server to find out which file is being requested. The PIC returns the file and

closes the socket.

With these subroutines, the PIC is now able to serve data to any web browser. Of course, this is of little use without an actual application defining what form of data needs to be handled. So without further delay, here are the applications I submitted to the PIC2000 contest.

## THE PIC WEB CAM

With the web server in place, it is easy to present web pages that contain textual data. Without images, this data is boring. When I was considering applications for the server,

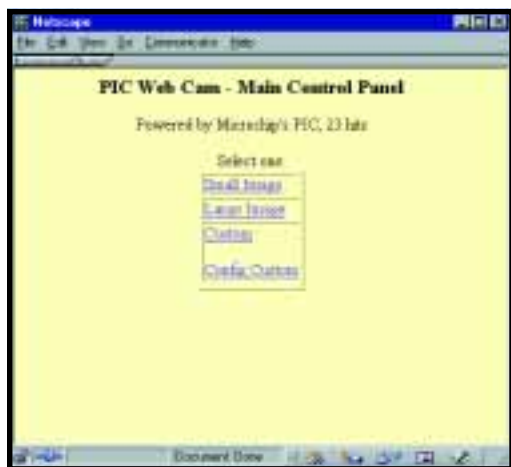
I asked myself "What better way to show the power of the PIC than an image being served up? Not only that, what if the image was live?" This led to the PIC Web Cam, which received grand prize in the PIC2000 contest (see Photo 1).

Several web servers on the Internet offer live images. They typically involve video cameras feeding image capture boards that are contained in large desktop machines with Ethernet boards. The PIC Web Cam accomplishes this with the web server. The images are obtained from an Olympus D-220L digital camera via its RS-232 interface. The image seen on the browser is refreshed automatically,



**Photo 2**—This is an example of the image presented by the PIC Web Cam's browser display. In this case, the image is automatically refreshed every 20 s, creating a live image.





**Photo 3**—Here's the PIC Web Cam main control panel as seen on the web browser.

creating a live image (see Photo 2).

The front of the sealed case is clear, allowing the unit to be used in harsh environments. I custom-built the yellow board above the camera that holds the PIC. The Ethernet board is located behind the camera.

This no-computer web camera is similar to the project published in *Circuit Cellar* 121, by Steve Freyder, David Helland, and Bruce Lightner.

They connected an inexpensive digital camera to their Picoweb server. The difference is their use of a Java applet to move the raw image from the server.

Because of the inherent image compression of the JPEG format, the PIC Web Server can deliver a higher resolution image in less time. In addition, you can use the PIC Web Cam with older web browsers that do not support Java.

After the PIC Web Cam is connected to the local Ethernet network, you can communicate with it by using ping or by requesting the default home page with the URL `http://IP_ADDRESS`, where *IP\_ADDRESS* is the PIC's preselected IP address. The resultant display on the web browser is shown in Photo 3.

Clicking Small Image decreases the image to  $160 \times 120$ , which is refreshed every 10 s. Large Image automatically reloads every 30 s. Lastly, Custom brings up the page that you can cus-

tomize via the Config Custom link (a form pops up with the existing HTML code). You can edit and store the code in nonvolatile memory.

On the custom page, parameters such as background color, image size, and refresh interval can be adjusted. Because the file is stored in the PIC's EEPROM, a maximum of 256 bytes can be stored.

Camera data is sent to the PIC in 2-KB chunks at 57,600 bps. The data is split because this is larger than one TCP/IP packet. Multiple groups of 2-KB data are transferred from the camera to the browser until the entire image is complete. It takes about 3 s to transfer a small image and about 6 s for a large image. The PIC web server has been tested with three simultaneous web clients.

This example illustrates the amazing capability of a tiny PIC processor. It can not only do the work of a desktop computer, but also present digital images from a still camera. The PIC processor is a valuable addition to any system requiring an inexpensive, effec-

tive way to transmit live images over the Internet.

## PIC WEB CHART RECORDER

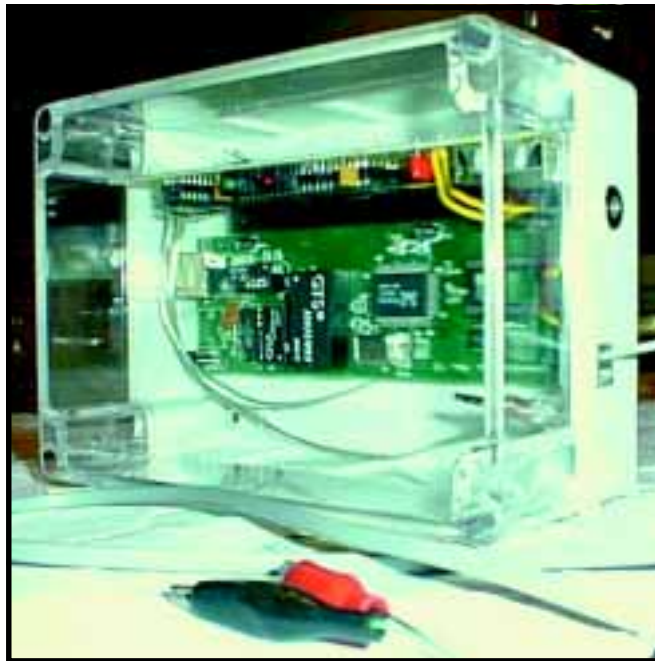
Another valuable project is the PIC Web Chart Recorder, which was awarded first prize in the Internet Applications category of the contest.

The graphical capabilities of the Internet provide a good way to present continuously varying data in the form of a chart. Sites including stock brokerages and weather stations benefit from this technology. Instead of using a large desktop computer, the PIC Web Chart Recorder (see Photo 4) provides this capability with a PIC.

Analog data is gathered by the PIC's A/D converter and presented by the generic web server. Use the control panel to control the sampling speed—stopped (no sampling), 10 s, 1 min., 15 min., or 1 h per

division.

The graphical data is packaged as a GIF image that is generated on the fly by the PIC. To simplify the generic



**Photo 4**—I configured the PIC Web Server as a Web Chart Recorder. The RJ-11 jack on the right of the box allows connection of the two alligator leads in the foreground. You would connect this pair to the data to be sampled.

web server code, it is split into five  $255 \times 52$  GIFs that are assembled at the browser. Because they are all the same width, they stack seamlessly. The display is automatically refreshed once per minute. During each refresh the graph is shifted from right to left to make room for new data.

The core C code that generates the GIF file is a modified version of the one published in *Circuit Cellar* 107, by Paul Breed. Modifications include changing it to a RAM-limited 8-bit environment and making it stream output into the Ethernet card.

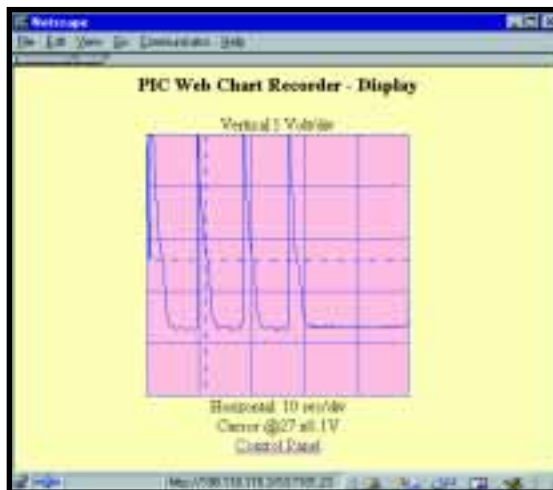
For better readability of the graph, horizontal and vertical dotted lines are added during creation of the GIF. In addition, where the data jumps by greater than one count from one sample to the next,

vertical lines are added to connect the graph in a continuous manner. Each column of pixels in the image represents one sample of the PIC A/D converter, and the vertical row represents the data value. As a result, the GIF is 256 pixels high (representing the one byte data range), and 255 pixels wide (representing the 255 samples that are stored).

The chart is defined as an HTML image map, causing the coordinates of the mouse cursor to be continuously displayed at the bottom of the web page. The browser updates the information as you move the cursor over the chart. This is useful when reading data values.

If you click on the GIF, a new image will be fetched from the PIC Chart Recorder, showing a dashed crosshair at the x-coordinate of the mouse click and the y-coordinate of the corresponding data value. In addition, the display underneath the GIF is updated with the voltage value of the sample. In Photo 5's example, the data value is 2.7 V.

The PIC Web Chart Recorder is a useful addition to any system where analog data needs to be gathered in a graphical form. The unit can not only gather data, but also present it in a web-friendly format. Real-time generation of GIF images, normally accomplished by desktop computers, is



**Photo 5**—The chart is presented in the form of a GIF that is generated on the fly in the PIC Web Server. Note the following: the data is the solid line; division markers are dotted lines. Moving over the GIF updates the mouse coordinates at the bottom of the web page (here they are 181, 23). Clicking on the image puts a dashed crosshair showing the value of the chart at the x-coordinate (its voltage is at the bottom (here it's 2.7 V)).

done here by a PIC.

Although the PIC Web Cam delivers data that is more interesting, the Chart Recorder represented a bigger challenge concerning software development. I did not know until the end if a complete web server and GIF file generator would fit in 8 KB of program space. Fortunately, the results exceeded my expectations.

In addition to the complex chart discussed here, the GIF can include objects that are easily described in mathematical form, such as circles, lines, and rectangles. So, the PIC Web Server can generate dynamic graphics such as thermometer bars, simple animation, and other indicators that change appearance based on real-time data acquired by the PIC.

## PIC X-10 WEB SERVER

For the next web server project, I went back to the original application that I had intended for the PIC Web Server, home automation. The home automation industry is dominated by a standard known as X-10. X-10 units communicate over the existing power line wiring to control all sorts of

loads. A simple power line modem enables a computer to send and receive these commands.

The PIC X-10 Web Server combines a server and an X-10 controller. With this project, you can control and monitor the status of your home or office over the Internet. First prize in the Internet Connectivity category was awarded to this project.

The X-10 Web Server control panel page automatically reloads itself every 5 s, allowing you to view the status of the units in a dynamic color-coded fashion. In addition, you can click on the hyperlinks to send commands to control the loads. If any other controller sends a command to the load, its status is also automatically updated in the home control panel.

The control panel can be reconfigured if you click on the Config button at the bottom of the main control panel. To add a unit, simply enter the name and X-10 address and click Add. You are then returned to the main autoloading home control panel with the new unit added (its status will be displayed as a blue question mark until a command is sent or received). Conversely, you can eliminate a unit from the control panel with Delete. The names and statuses of up to 16 units are preserved in the PIC's nonvolatile EEPROM.

The TW523 Two-Way X-10 power line modem uses a low-level, timing-critical interface. Thanks to the buffer in the NE2000 card, no particular attention needs to be paid to the server's Ethernet timing. An interrupt that fires with every power line transition ensures that bits get sent and received properly over the power line.

This powerful and compact web server will benefit any environment requiring the capability to send and receive X-10 commands without installing unusual hardware or software. Any computer on the local network can perform as the control panel.

## THE PIC SLIP SERVER

In addition to the previous two projects, a third design was successful



**Photo 6**—Many people can play tic-tac-toe against the computer simultaneously. The PIC won this time.

in the Internet Connectivity category of the contest. The PIC SLIP (Serial Line Internet Protocol) Server earned third prize.

Internet-connected PIC processors have thus far used serial ports for physical connection to the outside world. Unfortunately, the advantages of the PIC's small size are tainted by the need for a desktop computer as the Internet gateway. The SLIP Server addresses this deficiency.

The PIC processor's serial port provides the SLIP line. [7] The server allows almost any microprocessor to access the local IP resources (web and FTP servers, etc.) using only its serial port.

The versatile SLIP Server has many applications, among them is providing a wireless LAN connection. Using off-the-shelf wireless RS-232 links, you can put together a wireless network connection. A second application is establishing connectivity for computers with a serial port and no network card, such as PDAs, palmtops, and embedded computers. By adding a standard modem, the SLIP Server can create a remote access server. And it can provide Internet access for various other PIC projects, allowing them to send and receive data between FTP and web servers using only its serial port.

Using dial-up networking software, a Windows95 PC served as the SLIP client while I developed this project. The PC serial port and the SLIP Server were connected, and standard Internet clients (like Netscape and ping) were used to test the SLIP Server.

The main difference between IP data sent on the serial line and Ethernet is the Ethernet header (a.k.a., MAC). The latter contains the 6-byte hardware address of the source and destination nodes. Before any Ethernet data can be sent, the hardware address of the recipient must be determined by the Address Resolution Protocol (ARP). ARP requests are sent in broadcast mode, and the node that matches the addressing information responds with an ARP reply. Data obtained via the ARP protocol is stored in nonvolatile EEPROM memory. The data can be retained between power cycles or cleared during powerup.

Serial characters from the SLIP client are examined for the modem "AT...<CR>" character sequence. If one is detected, an "OK" string is echoed. This fools the PC into thinking that the modem has dialed and connected to the remote computer. The PIC continuously searches for the SLIP flag character, which demarks the start and stop of all packets. If it is at the beginning of a packet, the NE2000's registers are readied for loading data. If the flag character marks the end of a packet, the packet is examined to determine if the destination IP address is in the PIC's ARP table. If the entry is not found, an ARP request is sent to obtain the physical address. If the entry is found, the correct physical address is prepended to the IP packet and sent out on the Ethernet network.

On the Ethernet receiving end, new packets are checked for their type. A correct ARP reply is stored in the PIC's ARP table. If a remote IP node is querying the PIC for its hardware address, an ARP reply is sent back. Incoming IP packets are forwarded to the serial line with the correct SLIP flag and escape sequences.

In operation, the SLIP Server allows the client PC to access any IP service on the network. It also allows remote nodes to access any server on the client PC, showing that the SLIP Server has bridged the connection to the Ethernet. The client computer can now interact with the network resources as if it has its own Ethernet connection.

## PIC WEB TIC-TAC-TOE

Although this last web application hosted by the PIC did not win a prize, it is worthy of mention. When I was a young boy, I visited a computer display at a science center. A refrigerator-sized computer was located in the middle of the room attached to six large displays with keyboards. People could get a sense of the "intelligence" of the computer by playing tic-tac-toe against it. I was impressed that a computer could play against humans, and that it was difficult to beat.

This capability has been shrunk into a PIC processor running the web

server. Using any web browser on the local network, you can match wits with the PIC by playing against it over the Internet. As many as several dozen people can play simultaneously.

You can choose who makes the first move. An example game board during play is shown in Photo 6. During each turn, you move by selecting a hyperlink on an open square. This causes the web browser to retrieve a web page from the PIC. The returned HTML file contains a new game board updated with your move and the PIC's countermove. When either party wins, the winning row is highlighted in red.

The state of the game is kept in the hyperlinks. Thus, an unlimited number of people can play against the PIC because no additional storage is needed per player. The only limitation is the maximum number of moves that the PIC can handle per second. Assuming a conservative value of 20 moves per second and a 2-s pause between moves, the PIC can handle 40 simultaneous players! As a side benefit, because the entire state of the game is encoded in the hyperlink, you can bookmark Resume Game.

The strategy that the PIC uses to countermove is simple:

- if a win is imminent, go for the win
- if the opponent is about to win, go for the block
- if the center square is open, take it
- if a corner square is open, take it
- play any open square

This sufficiently simulates intelligent play for most matches.

The tic-tac-toe project illustrates how far silicon has come in the past few decades. Instead of a refrigerator-sized machine, the computer is the size of a fingernail. Teamed with a standard Ethernet card and attached to the local network, many players can match wits against the PIC using only their browser software.

## THE END RESULT

What happens when you combine a powerful little microprocessor with the ubiquity of the 'Net? You get the PIC Web Server. With the HTML language, a PIC can now communicate in



style with any user on the network. Complex and fancy user interfaces with control buttons, live images, dynamic graphics, charts, and up-to-date data are now possible. A single web page can contain data from an almost unlimited number of PIC Web Servers, making the display come alive with real-world data. 📡

*Edward Cheung works at NASA Goddard Space Flight Center in Maryland where he builds instruments for the Hubble Space Telescope. Astronauts install this hardware in space every two to three years. His interests include home automation and gardening. You can reach him at [edward.b.cheung.1@gsfc.nasa.gov](mailto:edward.b.cheung.1@gsfc.nasa.gov) or visit [cheung.place.cc](http://cheung.place.cc).*

## SOFTWARE

You can download the source code from [www.mindspring.com/~dr\\_ed/awards/pic2k/pic2k.htm](http://www.mindspring.com/~dr_ed/awards/pic2k/pic2k.htm).

## REFERENCES

- [1] Steve Freyder, David Helland, Bruce Lightner, "A \$25 Web Server," *Circuit Cellar* Online, July 1999, [www.chipcenter.com/circuitcellar/july99/c79b11.htm](http://www.chipcenter.com/circuitcellar/july99/c79b11.htm).
- [2] National Semiconductor, AT/LANTIC Ethernet chip, [www.national.com/ds/DP/DP83905.pdf](http://www.national.com/ds/DP/DP83905.pdf).
- [3] University of Aberdeen, IP packet structure, [www.erg.abdn.ac.uk/users/gorry/eg3561/inet-pages/ip-packet.html](http://www.erg.abdn.ac.uk/users/gorry/eg3561/inet-pages/ip-packet.html).
- [4] ARP packet structure, [melb.alexia.net.au/~www/vendor/internetinfo/arp.html](http://melb.alexia.net.au/~www/vendor/internetinfo/arp.html).
- [5] Rensselaer Polytechnic Institute, TCP checksum, [www.ecse.rpi.edu/Homework/shivkuma/teaching/sp99/i07-udp/sld015.htm](http://www.ecse.rpi.edu/Homework/shivkuma/teaching/sp99/i07-udp/sld015.htm).
- [6] University of Massachusetts, HTTP protocol, [gaia.cs.umass.edu/kurose/apps/http.htm](http://gaia.cs.umass.edu/kurose/apps/http.htm).
- [7] The Ohio State University, SLIP protocol, [www.cis.ohio-state.edu/htbin/rfc/rfc1055.html](http://www.cis.ohio-state.edu/htbin/rfc/rfc1055.html).

## RESOURCES

### Camera protocol

[www.average.org.digicam/protocol.html](http://www.average.org.digicam/protocol.html)

### NBC weather charts

[www.aws.com/nbc/wrc](http://www.aws.com/nbc/wrc)

### X-10 home control module

X10, Inc.

[www.x10.com](http://www.x10.com)

## SOURCES

### C Compiler

Custom Computer Services, Inc.

(262) 797-0455

Fax: (262) 797-0459

[www.ccsinfo.com](http://www.ccsinfo.com)

### MAX233

Maxim Integrated Products

(408) 737-7600

Fax: (408) 737-7194

[www.maxim-ic.com](http://www.maxim-ic.com)

### PIC 16F877

Microchip Technology Inc.

(480) 786-7200

Fax: (480) 899-9210

[www.microchip.com](http://www.microchip.com)