CIRCUIT CELLAR®
THE MAGAZINE FOR COMPUTER APPLICATIONS

by Fred Eady

Radio Roundup

Fred spent an entire month accumulating and categorizing embedded radio equipment. Has he become obsessed with collecting RF technology? It's a bit too early to label him a certified radiophile, *so give him a chance to explain what he's been up to. Who knows, perhaps his explanation will inspire you to incorporate some wireless gadgetry in your next design.*

It's time to work some more RF magic. So get that pointy wizard cap out of the closet and fetch that magic wand hidden in your dial caliper case. You may want to pull out your Western-style Stetson, too, if you have one.

You already know that the wizard gear is necessary to work with RF. But what do a wizard's cap, a sorcerer's wand, and a Stetson cowboy hat have in common? Well, I associate RF with the word "radio." And, being a 1950s baby, a big old Stetson hat is synonymous with the word "cowboy." When cowboys get together and bring along horses and cows to poke in a competitive environment, it's called a "rodeo." I don't have any cows and bulls to poke, and I didn't ride into town on a horse, but for the past month or so I've been ridin' the range and have managed to corral a few embedded data radios along the way. Now that you cowpokes are present, we can saddle up on our microcontrollers, put on our cowboy and wizard hats, and rope, ride, and wrangle some radios in the *Circuit Cellar* embedded radio rodeo.

I pulled together a collection of embedded radio equipment from various manufacturers to give you an idea of what's available these days, and I'll point out the features contained within the new embedded RF technology. A collective representation of all of the radio rodeo participants is shown in Photo 1. Now let's take a look at all of the embedded radio goodies I rounded up in the Florida room (in alphabetical order).

ABACOM XTR-903-A4

The Abacom Technologies XTR-903-A4's claim to fame is based on its size, ease of use, and programmability. Measuring in at 23 mm × 33 mm, it is almost as thin (5.5 mm top to bottom, excluding the mounting pins) as the PCB it's built on. The digital heart of the XTR-903-A4 is an Atmel ATmega8(L) microcontroller, which allows the XTR-903-A4 to communicate with an embedded host using simple TTL RS-232 signaling. All of the communication protocol stuff you would normally have to write is already coded into the XTR-903-A4's microcontroller. My XTR-903-A4 is a 433-MHz model, but you can get XTR-903-A4s that operate at 868 and 900 MHz.

The XTR-903-A4 operates using frequency modulation (GFSK) at 9600, 19,200, or 38,400 bps, with each data rate range having its own encoding scheme. Hamming and Manchester encoding are present when the XTR-903-A4 is used at 9600 bps. Manchester encoding alone is used when the XTR-903-A4 data rate is increased to 19,200 bps and a scrambling technique is used for 38,400-bps radio links. If your application doesn't need the speed, the 9600 bps radio link is optimal because it provides a higher level of data protection and longer range (up to 200 m in open air with an omnidirectional antenna).

The data rates are selectable using the XTR-903-A4's SP1 and SP2 control pins. The pins can be tied permanently or controlled via a host microcontroller.

While I'm on the subject of configuration pins, the XTR-903-A4 can be put into Power Down mode by the

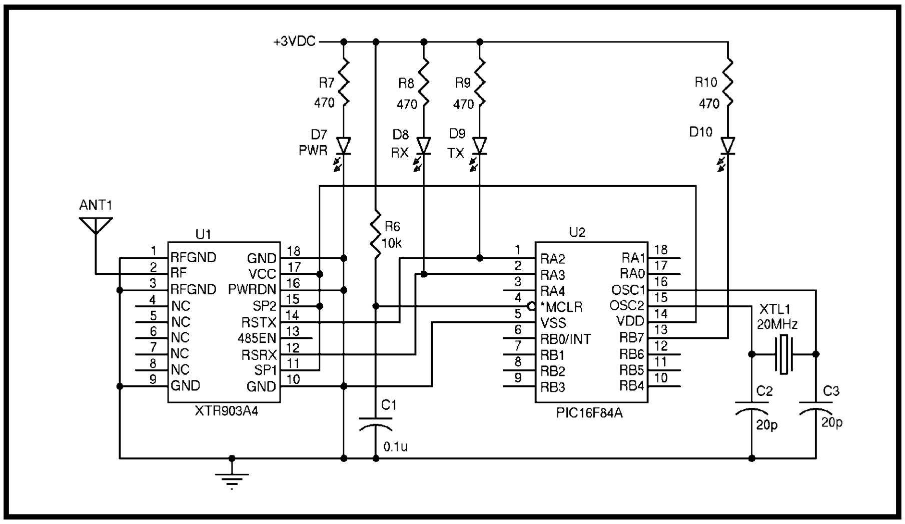

+3VDC
R7 470
R8 470
R9 470
R10 470
D7 PWR
D8 RX
D9 TX
D10
ANT1
U1
RFGND RF RFGND NC NC NC NC NC GND
GND VCC PWRDN SP2 RSTX 485EN RSRX SP1 GND
XTR903A4
R6 10k
C1 0.1u
U2
RA2 RA3 RA4 *MCLR VSS RB0/INT RB1 RB2 RB3
RA1 RA0 OSC1 OSC2 VDD RB7 RB6 RB5 RB4
PIC16F84A
XTL1 20MHz
C2 20p
C3 20p

Figure 1—*The XTR-903-A4 hardware interface is dead easy because the blinking lights are optional.*

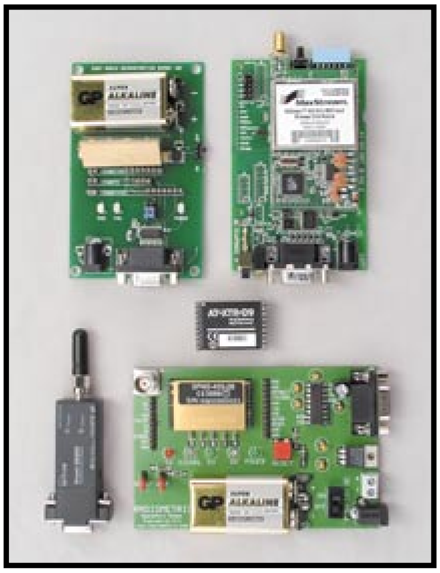


Photo 1—*Here's a glimpse of all of the radio participants. Each data radio has its own personality. That's a good thing because having a variety of differing features allows you to select the right radio for your application.*

host microcontroller using the PWRDN pin. The XTR-903-A4 draws just under 10 µA in this mode.

To keep things really simple, the XTR-903-A4 uses a set of unique AT commands to select one of 10 channels between 433 and 434 MHz and set the module's RF output power. An AT command to monitor channel activity and signal strength is also incorporated.

XTR-903-A4 transmission and reception synchronization is automatic thanks to the embedded ATmega8(L). All you have to do is feed data to the XTR-903-A4's serial interface without any regard for data length. The XTR-903-A4 doesn't do any data buffering, nor does it add a CRC or any other form of checksum data to the transmitted data. A preamble is automatically inserted at the beginning of a transmission to allow the remote XTR-903-A4's receiver to synchronize. The inclusion of the preamble induces a data transmission latency of about 20 ms, which means you should allow 20 ms for transmit-and-receive switching in your application code. The logic inside the XTR-903-A4 also automatically appends an end-of-transmission data sequence to the transmitted data packet.

My set of XTR-903-A4 embedded radios didn't come with an evaluation board. So, using the straightforward technical detail included in the XTR-903-A4 documentation, I easily crafted my own set of XTR-903-A4 evaluation boards (see Photo 2 and Figure 1).

I selected the PIC16F84A as the host microcontroller because it doesn't require a lot of effort to employ, can operate at the 3-VDC level required by the XTR-903-A4, and is superbly supported on the programming side by PICSTART Plus, MPLAB, and the CCS C compiler. As you can see in Listing 1, the inclusion of the Atmel microcontroller in the XTR-903-A4 circuitry makes it easy to code host microcontoller transmit and receive routines.

MaxStream 9XStream

My first impression of the 9XStream came when my UPS driver handed me the 9XStream evaluation kit box. It was a big box, and I figured it held either a complicated data radio kit or lots of goodies to support a simple, well-equipped embedded data radio rig. Before installing the CD-ROM-based documentation and configuration program, I rummaged through the evaluation kit to see if I was going to have to go the Florida

Photo 2—*I used the PCB construction detail contained within the XTR-903-A4 documentation to fabricate this evaluation board. What you don't see are all of the SMT transistors, resistors, and capacitors that support the status LEDs.*

Listing 1—*A simple RS-232-capable microcontroller and some just as simple code are all it takes to put the XTR-903-A4 on the air.*

```
#include <16F84A.h>
#use delay(clock=20000000)
#fuses NOWDT,HS, PUT, NOPROTECT
#use
rs232(baud=9600,parity=N,xmit=PIN_A2,rcv=PIN_A3,bits=8,stream=to_
radio)

#byte PORTB = 0x06
// Ping-pongs 0x41 between unit 1 and unit 2, and complements
port B on both units with each transmission.
void main()
{
// Common code for both units
   int8 data;
   set_tris_a(0xFB);
   set_tris_b(0x00);
// Unit 1 code
   while(1)
   {
    putc(0x41);
    while(!kbhit());
    data = getc();
    if(data == 0x41)
     PORTB = ~PORTB;
   }
// Unit 2 code
   while(1)
   {
    while(!kbhit());
    data = getc();
    if(data == 0x41)
    {
     PORTB = ~PORTB;
     delay_ms(1000);
     putc(0x41);
    }
   }

}
```

room library and meditate with the 9XStream documentation.

Three of what seemed to be communications interface adapters caught my eye. The first adapter pulled from the shipping foam was pink and imprinted with the word "LOOPBACK." Heck, I know what a loopback adapter is. No worries here, I assured myself.

Then, I asked myself what I'd do with a loopback adapter in a wireless network. OK. Next adapter. This black adapter was labeled "Null Modem Adapter." I'm batting a perfect 1000 because I can also tell you what a null modem adapter is used for. Again, I asked myself what I'd do with one in this environment.

I shrugged off the self-inflicted questioning and moved on to the third and final adapter, which was white and identified as "Null Modem." The tag on the plastic bag also pointed out that this adapter was a female-to-female null modem adapter. Hmm. I pulled out the black null modem adapter again and noted that both of its interfaces were male. As you can imagine, I was ignorant at this point, and this group of out-of-place adapters really had me swinging. What in the world am I going to be looping back? I wondered. What am I going to do with "null modeming" in a set of wireless data radio modules?

Fortunately, the rest of the parts and pieces made a bit more sense to me. The following were easily identifiable: a couple of nine-pin male-to-female signal cables for attaching to PC serial ports, two 9-VDC wall warts, and a pair of male and female nine-pin to RJ-45 adapters to allow the use of Cat 5 cabling as data cable. The neatest cable I found adapted a standard 9-V battery to the wall wart power receptacle on the evaluation/development board. I figured that the examination of the actual data radio electronics would give me some clues as to what to do with those crazy communications adapters.

I decided to begin by examining a 9XStream evaluation board. The RS-232/485 circuitry looked standard enough, as did all of the pinouts for interfacing to the actual 9XStream data radio module.

DIP switch settings determine if you have to incorporate parity or run RS-232, two-wire RS-485, or four-wire RS-485. 9XStream RS-485 termination options are also DIP switch-selectable. The remainder of components on the evaluation board consists of a low-power, low-dropout voltage regulator and supporting circuitry, a configuration push button, and an ATtiny26 microcontroller. Hmm.

The 9XStreams that I dug out of the box are compact data radios that operate in the 900-MHz band at 9600 bps. The 9XStream's RF circuitry is confined to a shielded enclosure with an ATmega32(L) being the most prominent part mounted outside the shield.

After I assembled the 9XStream radios and evaluation boards, I connected them to separate PCs in the Florida room. Instant gratification: the radios worked perfectly together right out of the box. The next step was to sort through the 9XStream documentation to try and figure out what to do with the adapters. The documentation on the CD-ROM is extremely detailed and complete.

Photo 3—*Here's an example of "drop in" RF. The EDTP Easy Ethernet ASIX board can now add wireless to its Ethernet and I²C connectivity resume. All I had to do to wireless-enable the Easy Ethernet board was change the data rate in the PIC firmware.*

The 9XStream can be programmed using 9XStream-unique AT-type commands or via the X-CTU program, which is on the CD-ROM. The AT commands that can be issued fall into four categories: diagnostic services, networking services, command mode operations, and serial interfacing. The X-CTU interface automates the AT command configuration process and includes extra services like range testing and an ASCII terminal interface.

While experimenting with the X-CTU application, I came across a section in the advanced 9XStream documentation that solved the communications adapter mystery. The pink loopback adapter is used to loop received data back into the transmitter during range testing. Using the pink loopback adapter converts the stationary remote radio into a repeater that simply echoes the data sent from the 9XStream radio you're walking away with during the range test. The black null modem adapter is used to connect the 9XStream's DCE interface to another target DCE interface. The white female-to-female null modem adapter replaces a set of 9XStream data radios and can be used to connect and test the DTE interface cables that connect the 9XStream to each respective host.

Now that everything has been identified and categorized, what does it take to connect the 9XStream to an embedded host? The answer is simple: three wires. From the 9Xstream's perspective, they are data in (DI) from the host, data out (DO) to the host, and clear to send (CTS). All else is automatic. The 9XStream adds a factory-assigned vendor ID number (VID), a channel number (ATHP), a module address (ATDT), and a packet serial number (PSN) that identifies each packet in front of the data received from the host. A 16-bit CRC is appended to the outgoing packet.

There are also 9XStream pins that provide optional features such as power-down, reset, and module status that can be used by the host and host application. The 9XStream has a useful range of up to 1500′ with a standard whip antenna. If your 9XStream radios are configured with special high-gain antennas, you can obtain signal ranges of up to 20 miles line of sight. Using the DCE-to-DCE adapter, I 9XStream-enabled an EDTP Easy Ethernet ASIX board, which happens to have an on-board PIC microcontroller supporting a simple three-wire RS-232 DCE interface (see Photo 3).

easy-Radio MODULE

While ridin' the range, I managed to rope a few easy-Radio data radios consisting of a 433-MHz pair, a 900-MHz pair, and a 900-MHz frequency-hopping pair. The 433- and 900-MHz modules are identical. The only way to tell the easy-Radio modules apart is to read their brand—ER400TRS for the 433-MHz

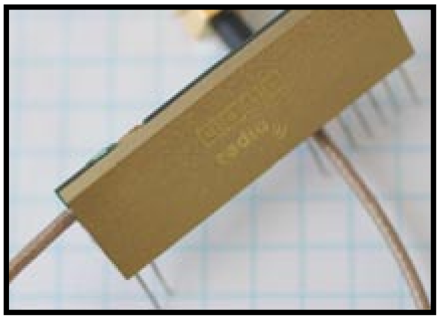


Photo 4—*The coax attachment feature is nice because you aren't tied down to placing an antenna connector pad on your final PCB.*

modules and ER900TRS for the 900-MHz modules. Both frequency ranges of the easy-Radio module pairs can be plugged into a common set of evaluation boards, which are included in the easy-Radio evaluation kit.

I managed to take the "easy" out of the new easy-Radio evaluation boards by making assumptions and not paying attention to the documentation. The frequency-hopping easy-Radio modules, which are marked as ER900FHTRS, are larger than the non-hoppers because of some extra pins on the hopper radios. If you're a *Circuit Cellar* regular familiar with your back issues, you know that I'm not new to easy-Radio technology ("RF Made Simple," *Circuit Cellar*, issue 160, November 2003).

In fact, some of the legacy easy-Radio equipment is still on the Florida room's shelf. The legacy and new easy-Radio evaluation boards provide a socket position that accepts the slightly larger hopper radios. Of course, if it fits, plug it in. Right?

Hours later, after trying all sorts of things, I could not get the new hoppers to talk to each other. I turned to the easy-Radio documentation out of desperation. I noticed that my new evaluation boards were void of the valid status LED. I unboxed my old down-level hopper evaluation boards and lo and behold there it was, the extra valid indicator LED. I started comparing the ER900FHTRS sockets on each evaluation board and found that the antenna pin for the legacy hopper modules on the new evaluation board had to be grounded, which is a subtle hint that you shouldn't be using the new evaluation board to test down-level hopper easy-Radio modules.

I can get around that, I thought, so I flexed what I surmised to be the new hopper's antenna pins to fit into the antenna pins on the smaller module socket behind the hopper socket. I could see the trace to the antenna from the smaller socket pin behind the hopper's socket, and I thought this would make things all better. Ha! Nothing changed. Still no signal.

Further investigation of the ER900FHTRS version 2.00 Quick Start Guide pointed out that what I thought to be the antenna pin on the new modules is indeed a ground pin. Aha! So that's what that pair of antenna coax leads are for.

As it turned out, the new ER900FHTRS hoppers have an antenna receptacle cut out in the side frame of the easy-Radio hopper module that accommodates the coax antenna lead that came with the easy-Radio hopper evaluation kit. What was not obvious to me will be obvious to you if you take a look at Photo 4.

The smaller 433-MHz easy-Radio modules plugged right in and worked right out of the box. If I had slowed down and done some reading, I would have had the same success with the easy-Radio frequency hopper modules. In addition to the new antenna connection, the new easy-Radio ER900FHTRS radios can operate in Client/Server mode at 38,400 bps. An improved version of the easy-Radio configuration software on CD-ROM also comes with the new evaluation kits.

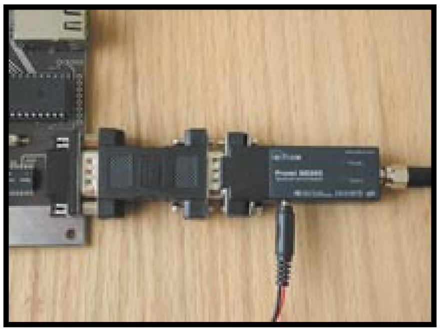
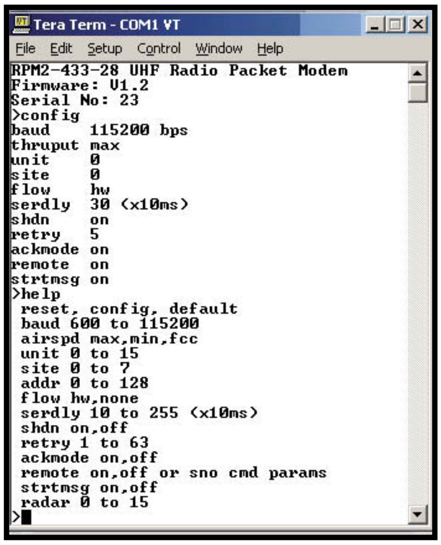


Photo 6—*The SD-202 gets it power from the EDTP Easy Ethernet ASIX via a DC power cable assembly that comes with the Bluetooth radios. Mark, the machinist that fabricates all of my weird metal and plastic stuff for my* Circuit Cellar *articles, is also using Bluetooth modules in his machine shop to communicate with a prototype project he is working on.*



```
RPM2-433-28 UHF Radio Packet Modem
Firmware: V1.2
Serial No: 23
>config
baud    115200 bps
thruput max
unit    0
site    0
flow    hw
serdly  30 (x10ms)
shdn    on
retry   5
ackmode on
remote  on
strtmsg on
>help
 reset, config, default
 baud 600 to 115200
 airspd max,min,fcc
 unit 0 to 15
 site 0 to 7
 addr 0 to 128
 flow hw,none
 serdly 10 to 255 (x10ms)
 shdn on,off
 retry 1 to 63
 ackmode on,off
 remote on,off or sno cmd params
 strtmsg on,off
 radar 0 to 15
>
```

Photo 5—*The SpacePort SPM2-433-28 is an international kind of data radio. Note the FCC airspeed setting. If you're wondering, the radar setting is for range testing.*

RADIOMETRIX SpacePort

Atmel has been well represented in the embedded radio controller role thus far. The Radiometrix SpacePort data radio prevents a shutout and incorporates a PIC16F876 to invoke its RF features. From what I could glean from the SpacePort documentation, the PIC is the radio packet modem (RPM). A fast radio packet controller (FRPC) works in conjunction with the RPM and resides under the shield with the SpacePort's RF circuitry. It's just a guess, but looking at the SpacePort documentation, the FRPC is most likely a PIC16F84A running at full speed (20 MHz).

My SpacePort module (SPM) evaluation kit came with a pair of SPM2-433-28 433-MHz radio modules and a pair of matching evaluation boards. Options can be exercised to get the SpacePort modules configured for the 869- and 900-MHz bands. There are a bunch of configuration options that can be dialed into the SpacePort, and the configuration parameters are set using a simple terminal program like HyperTerminal or Tera Term Pro. Photo 5 is a shot of what my SPM spit out when I jumpered it into Setup/Configuration mode.

As you might ascertain from the SPM configuration entries, the SpacePort is packet-oriented and designed to work well in addressable wireless networks. To that end, the SpacePort performs error checking, packet acknowledgement, and retrans-

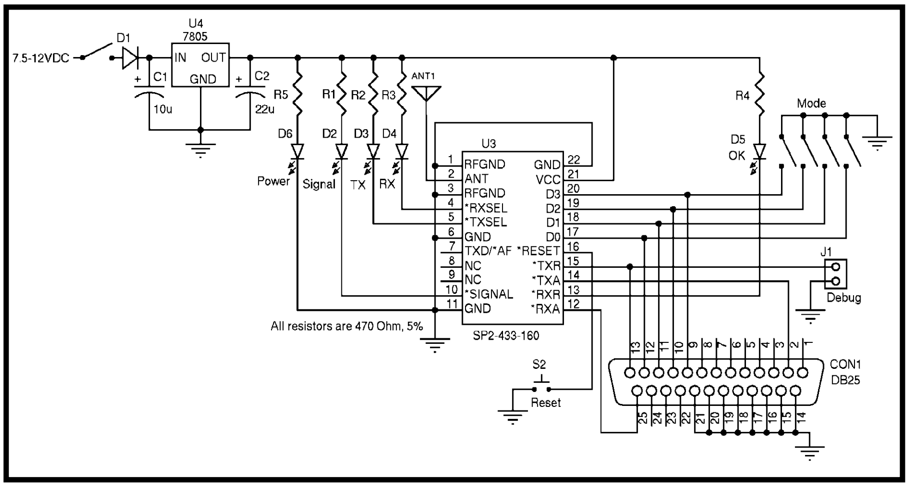


Figure 2—*To make implementing the SpacePort as simple as implementing the XTR-903-A4, you can use any microcontroller, eliminate the RS-232 driver, pull your 5 VDC from an existing power source in your project, and chuck the LEDs.*

mission functions to help guarantee the integrity of data flowing on the wireless links. The SpacePort's maximum acknowledged throughput is 28 kbps. Unacknowledged throughput is rated at a maximum of 55 kbps.

The SpacePort only requires a simple three-wire interface to a controlling host: transmit data (TXD), receive data (RXD), and clear-to-send (CTS). However, particular attention has been directed toward providing plenty of visual feedback relative to the condition of the wireless link and the data flowing across it. Figure 2 is a pretty good representation of how the SpacePort module can be deployed; it closely follows the way the SPM is implemented on the SpacePort evaluation board.

INITIUM

No, I haven't forgotten my ABCs. The Initium Promi-SD202 is a Bluetooth product, and I didn't want to mix the bulls with the broncos. I've been experimenting with some Bluetooth nodes and have used them to replace RS-232 cables between gadgets in the Florida room.

The Promi-SD202 is a class 1 Bluetooth transceiver powered by a 5-VDC power source that can be attained from a PC USB port, an optional wall wart, or a direct connection to the host power subsystem using a supplied DC power cable. It can operate at data rates ranging from 1200 to 230,400 bps, and it connects to the host via an integral nine-pin RS-232 DCE interface. The effective range is approximately 100 m.

Getting on the air with the Promi-SD202 modules was a breeze. A configuration program is included with the module pair that permits you to select data rate, connection, and security options. There are four modes of operation defined in the configuration program. Mode 0 is used to initially set up the SD202 modules, which involves selecting over the air data rates and discovering other Bluetooth modules that are connectable and within range.

I wanted the SD202 module pair to always connect at power-up, so I used the mode 0 functions to allow the modules to learn about each other (device address, device name, class of device, etc.) and establish an initial connection. Both SD202 modules automatically store the Bluetooth device address of its latest counterpart, which allowed me to set one module to mode 1 and the other to mode 2.

Mode 1 instructs the SD202 to connect only to the last connected device, while mode 2 tells the companion module to accept only a connection from the last device it was connected to. After the initial connection is complete and the modes are set, the Promi-SD202 modules can be used as 100-m virtual data cables between machines and various RS-232-equipped gadgets in the Florida room.

I'm always testing the RS-232 interfaces on the Easy Ethernet devices that I ship. Using Bluetooth instead of a cable removes the distance restriction when a unit goes back to the bench for testing. I Bluetooth-enabled an EDTP Easy Ethernet ASIX using the black male-to-male null modem adapter from the 9XStream evaluation kit (see Photo 6).

GIT ALONG LITTLE DOOGIES

We've ridden and roped every data radio that we've brought with us. Now it's time to gather all the livestock and move on. If every segment of technology worked as hard to make its products as easy to use and assimilate as the embedded data radio folks do, we would already have a colony on Mars.

All of the embedded data radio units I mentioned are designed to be dropped into your final product with a minimum of wand swinging. Visit each manufacturer's web site, and you'll find lots of additional information to help you incorporate wireless into your next design. Using them in your projects allows you to wear your pointy RF wizard hat to anybody's rodeo with pride because you will have helped to prove that RF doesn't have to be complicated to be embedded.

Fred Eady has more than 20 years of experience as a systems engineer. He has worked with computers and communication systems large and small, simple and complex. His forte is embedded-systems design and communications. Fred may be reached at fred@edtp.com.

SOURCES

XTR-903-A4 Transceiver module
Abacom Technologies
www.abacom-tech.com

EDTP Easy Ethernet ASIX board
EDTP Electronics
www.edtp.com

XStream Wireless development kit
MaxStream, Inc.
www.maxstream.net

easy-Radio
Low Power Radio Solutions Ltd.
www.lprs.co.uk

SpacePort and Promi-SD202
Lemos International Co., Inc.
www.lemosint.com

PICSTART Plus and MPLAB
Microchip Technology, Inc.
www.microchip.com