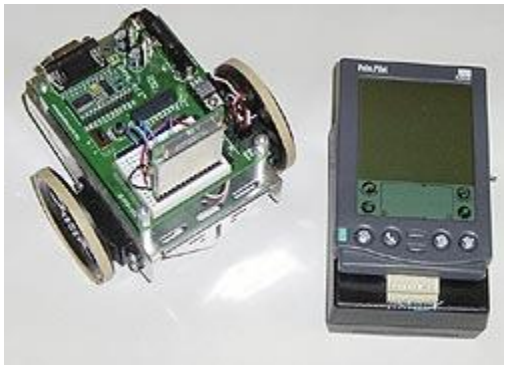
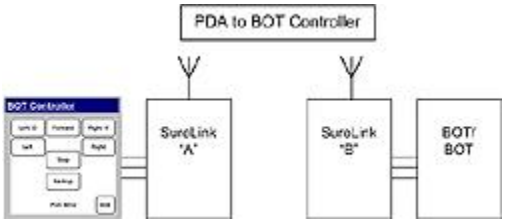
PDA-BOE/BOT Controller

By Mark Spencer, WA8SME
Former ARRL Education and Technology Coordinator



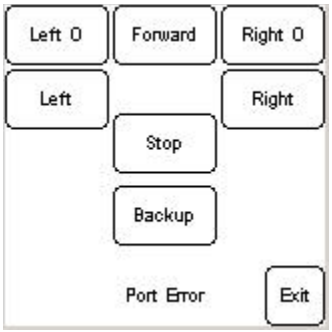
The purpose of this project is to give teachers some ideas that they can use to help their students to understand what happens in today's wireless link systems. Handheld wireless devices have hit the mainstream in a big way. Wireless linked cellular telephone systems and linked Personal Data Assistance (PDAs) and palm sized computers are becoming common place. But behind these unassuming technologies are some pretty sophisticated hardware and software. Through a project similar to the one described here, students can get a better appreciation of what they hold in their hands. The idea for this project was stimulated by a chapter written by Al Williams[1].



Overview. The PDA-BOE/BOT Controller is a relatively simple user interface that allows someone to control the movements of a BOE/BOT robot by pressing directional controls displayed on a PDA screen. The user commands are sent from the PDA to one side of a data link transceiver pair that transmits the commands from the PDA over radio frequencies to the receiving component on the BOE/BOT. The BOE/BOT accepts the commands and takes action. Sounds pretty simple, but in reality there is a lot of sophisticated stuff going on behind the scenes that makes this little control kind of elegant.

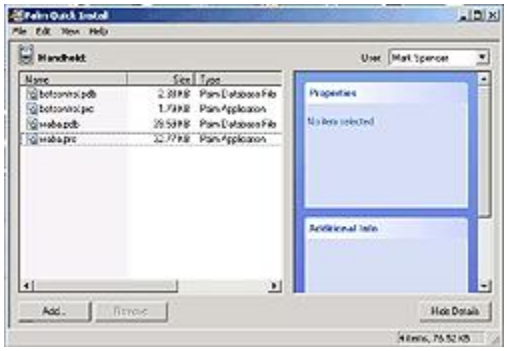
In this project, I used one of the first generation Palm Pilot PDAs. I used the Palm Pilot first because I had one and second, this project requires some simple mechanical and electrical (soldering) modifications so the older Palm Pilot was not too dear to experiment with. The datalink transceivers are SureLink 900MHz RF Modules (Parallax #30065 and companion programming board #30066). The circuit diagram of the few ancillary components needed to interface the individual parts of the controller accompanies this article.

Handheld PDAs. Of course the handheld devices are becoming more powerful with each increment of product improvement, however, the PDA is not yet capable of running full-up desktop software. The programming language used on the Palm is a derivative of the Object Oriented language JAVA[2] called WABA[3]. I will not attempt to cover the Palm software in detail, but I hope the reader's interest will be peaked enough to check out JAVA and WABA. The source code for the BOT Controller is available upon request[4] and hopefully the commenting in the source code can help the reader follow the logic of the programs.



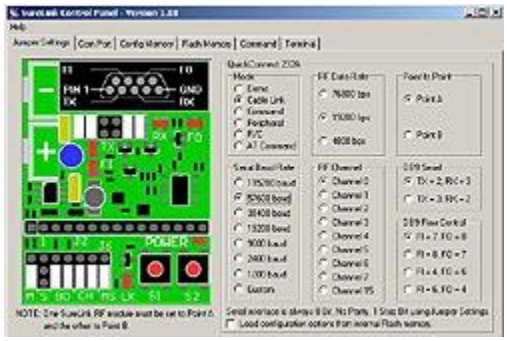
Basically the Palm software displays directional keys on the screen and opens the serial port of the Palm to the SureLink transceiver. When the user touches one of the control buttons, a command represented by a letter is placed in a transmission cue awaiting a signal from the BOE/BOT that it is ready to receive the command. Once the go-ahead is received, the Palm sends the command to the SureLink for transmission.

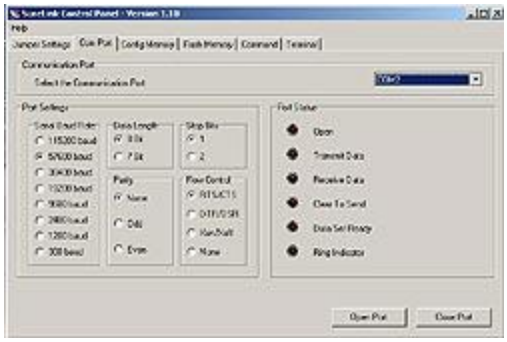
There are four files that have to be downloaded to your Palm (also available upon [4]request). Use your device specific installer program to download the files. The screen display of my installer program is illustrated here. Once installed, you will see two added icons to your Palm applications list. The one you will use for this project is the botcontrol.

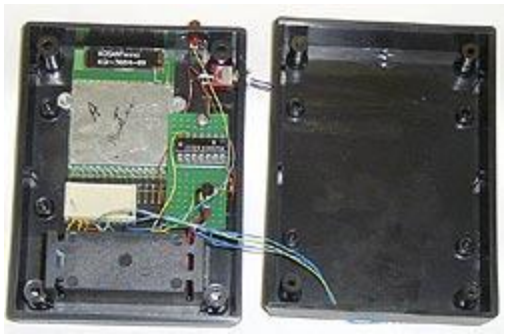


Wiring. You will need to find the serial pin configuration of your Palm in order to make connections between the Palm and the SureLink transceiver. The lines needed are the TX, RX, and ground lines. I used a voltmeter and the Palm's docking cradle to determine which pins were which. Once the pin configuration is determined, you can do the appropriate Palm modifications for your particular situation.

In my case I wired in and secured a 10-pin plug so that I could continue to interface the Palm to the docking cradle and also connect the Palm to the SureLink transceiver and supporting circuits.







SureLink. The SureLink[5] transceivers have a number of operating modes and are extremely flexible and capable systems. They use the 900 MHZ frequency range and advertise a maximum range of 1000 feet. The SureLink can operate in a cable link mode (serial replacement mode), AT Command mode, and RC Mode to name just a few of the most useful modes. In this project, the SureLinks use the cable link mode. (The reader also needs to check out the RC mode, way cool.) The SureLink can be configured using internal flash memory or can be configured using external wiring. I chose to use the internal flash memory to configure the systems to reduce the amount of interconnecting wiring. This scheme adds the complication of programming the SureLink units but it is worth learning.

To program the SureLinks, you will need the QuickLink Demo Board and the supplied SureLink_Control program. Install the SureLink into the demo board and connect the demo board to the computer's serial port. There is a panel in the software that helps you set the demo board's jumper settings. It will take some experimenting with the settings to open the COM port with the demo board. Finally, once you are talking to the demo board you're ready to program the SureLink.

I programmed the Palm transceiver to be unit "A" and the BOE/BOT transceiver to be unit "B". The serial port is set to 9600, 8-bits, no-parity, simplify the interfacing. The transmission baud rate is set at 19.2K just to ensure good datalink connection (higher baud rates reduce effective range). The other more fancy parameters are left at default values. A screen shot of the Config Memory panel of the SureLink_Control Panel for programming one of the SureLinks is shown here.

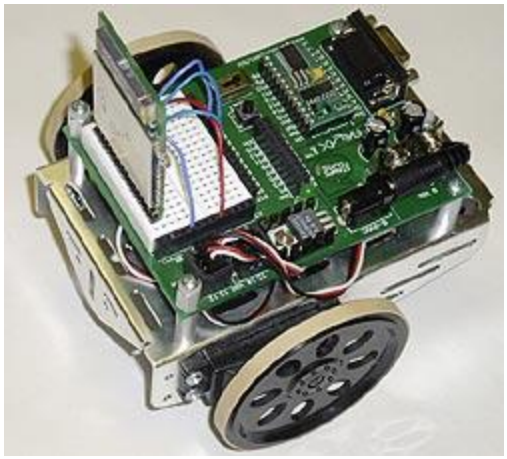
Once the SureLink is programmed to stand alone, as long as pin 1 is left floating during power-up, the SureLink will be configured using the flash memory.

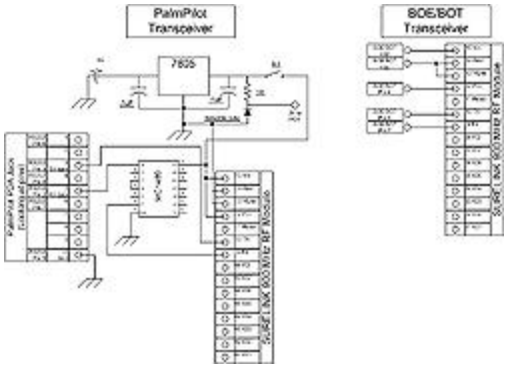
There is one other complicating factor. The Palm sends serial data using RS232 protocol, the SureLink uses TTL levels. The added RS-232 receiver IC takes care of the voltage translations. The channels of this particular RS-232 receiver are inverting, so the signal is sent through a second receiver to produce right-side-up TTL signals the SureLink can understand.

My Palm Pilot has a tendency to eat batteries so I added a power source for the Palm that uses larger, and longer lasting batteries than the AAA cells included in the unit. The SureLink needs a 5-volt source anyway; it was a simple matter to tap off a little current for the Palm.

The SureLink on the BOE/BOT side is easy to interface using the proto-board of the BOT and the V_{dd} and V_{ss} connections. The TX and RX lines of the SureLink are connected to appropriate pins on the BStamp of the BOT. One additional line is needed for flow control of the data being sent from the SureLink. A high on the FI/pin 12 of the SureLink allows the data stored in the serial buffer to be transmitted. The programming in the BOT switches this line low-to-high-to-low at the appropriate time to control the follow of data.

BOE/BOT. The software in the BOT is set up to simply send a signal (the letter "k") to the Palm to indicate that it is ready to receive a command. The user-supplied command is sent to the BOT, and the software jumps to the appropriate movement routine to move the robot. The BStamp has very limited interrupt service capabilities and therefore the BOT can only do one thing at a time, either move, or wait and accept a new command. This requires some handshaking between the BOT and the Palm.





When the BOT receives one of the turning, stop, or backup commands, the BOT executes the movement and then stops and asks for the next command. If no command is forthcoming, the BOT continues to wait. If the BOT is moving forward, it will move forward for approximately 10 inches, stop, send a prompt to the Palm, and if no command is received, the BOT will continue forward an additional increment. If an alternative command is received, that command is executed and the BOT waits.

Again, hopefully the commenting on the BOT software will help the reader follow the programming logic.

Simple, yet not so simple. So what are the instructional concepts behind this little project? A partial list includes:

- Java/Waba programming
- User interface strategies
- RS-232 to TTL voltage conversion
- Serial communication
- RF transmission ranges and wave propagation
- Microprocessor programming
- I/O strategies
- Power management
- Data flow control
- Etc., etc, etc.

There is a lot going on in this little project. Perhaps doing this kind of project will give the student a better appreciation for the sophistication and elegance behind handheld wireless technology.

Notes:

[1] *Al's "World Famous" Stamp Project of the Month Anthology*, Al Williams, Parallax Inc., Pages 177-183.
[2] http://java.sun.com/
[3] http://www.wabasoft.com/
[4] etp@arrl.org
[5] http://www.parallax.com/