BlueSense
Final Report

Hypheng Lim
hypheng@gwu.edu

CSCI297

2007-04-24

Project Abstract

The goal of this project is to establish a wireless bluetooth communications channel between the Zilog Z8 microcontroller and a Sony Ericsson P910i mobile phone. Once connected, the P910i would be able to send commands to the Z8 to request certain data, such as readings from a temperature sensor. Additionally, the P910i can send data messages to the Z8 to be displayed on a connected LCD device.

The P910i supports bluetooth natively as well as the required extensions for java bluetooth support (JSR82). The Z8 will be connected to a BlueSMiRF bluetooth modem from Sparkfun.

The proposed requirements for this project are:
- Establish a wireless bluetooth connection between the P910i and the Z8
- The P910i must allow the user to enter and send commands/data to the Z8.
- The Z8 will read and parse commands sent from the P910i.
- The Z8 will send sensor readings when a specific command is sent.
- The P910i will read and parse responses from the Z8 for display to the user.

Status

Major functions all work as intended.

Changes to initial design:

- Used LCD display as opposed to LED array on the Z8. The LCD display does not require scanning to draw various characters. In addition, it displays more characters at once (16 vs. 4).
- Software flow modified to support initial setup of bluetooth.

Specification

Description

There are two major devices for this project, the Sony Ericcson P910i mobile phone and the Zilog Z8 Encore! Evaluation Board. The Sony Ericcson P910i has built in support for bluetooth and does not require any external hardware. A J2ME client program will be written for the P910i. The Zilog Z8 has several components in addition to the evaluation board. It will use the BlueSMiRF from Sparkfun for bluetooth communications. In addition, an LCD display and temperature sensor will be connected. A C server program will be written to drive its various external components as well as handle communications with the P910i.

Hardware Modules

Sony Ericsson P910i has the following characteristics related to this project:

- GSM mobile phone
- CLDC1.0/MIDP2.0 J2ME support.
- JSR 82/bluetooth support within J2ME.
- Input via touch sensitive screen or attached QWERTY keyboard.
- 64MB built in storage space

Technically, any phone that meets the above requirements should work with the client program. However, only the P910i has been tested. Additionally, some phones may require more setup then was required by the P910i.

Zilog Z8 Encore! Evaluation Board

- Z8F6403 8 bit microprocessor
- UART/RS232 interface for BlueSMiRF communications
- 80 GPIO pins (8 for LCD driver)
- I2C interface for temperature sensor
- Part#: 99C0868-01

AD7414-2 I2C Temperature Sensor

- 10 bit temperature to digital converter
- Standard I2C serial bus support.
- -40 to 125 degrees C temperature range

MDLS 16166-01 16x1 LCD display device

- Standard HD44780 Display
- 8x2 matrix arranged as a 16x1 display
- 4 and 8 bit data interfaces (4 is used).

BlueSMiRF v1.0 Bluetooth Modem

- UART interface, full duplex up to 115200bps
- Built in bluetooth stack
- 200-300ft range
- 9600-8-N-1 default port settings

Software Development Tools

Java JDK 1.5.0_11

Sun Java compiler. It is used to compile the client program for the P910i mobile phone.

Sun Wireless Tool Kit (WTK) 2.5

This tool kit will compile, package and sign applications ready for deployment to mobile devices. In addition, it has an emulator which helped with debugging the interface.

Sony Ericcsson PC Suite 3.1.1 (optional)

This program allows the user to install programs to the P910i via a USB cable. Alternatively, programs can be loaded via Bluetooth (file transfer from PC) and then installed by the phone.

Eclipse 3.2, EclipseME 1.6.6 (both optional)

These two programs are an integrated development environment for Java. EclipseME is a plugin for Eclipse to aid in developing J2ME programs. They were both used mainly for editing the Java programs. The Sun WTK was used to compile and package the application. Alternatively, any simple text editor will suffice for development. Eclipse makes it a bit easier with various features such as code refactor and real time error checking (static).

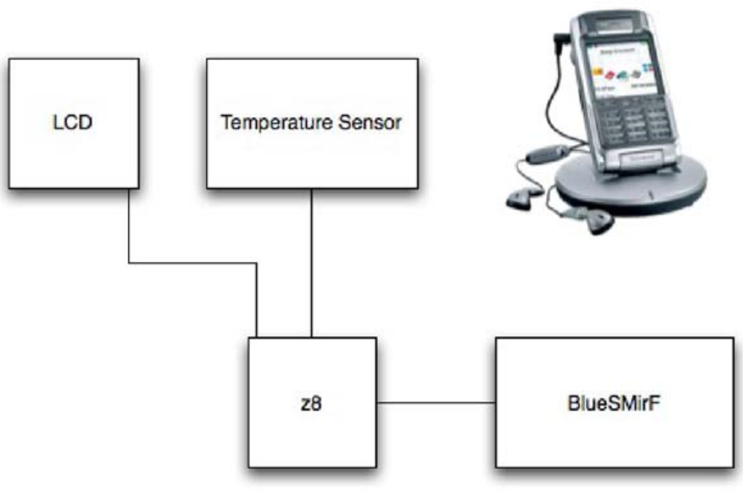
Zilog Z8 Encore! Family IDE version 4.10 (06121401)

Used to develop and debug the server side program which runs on the Zilog Z8.

Implementation & Construction

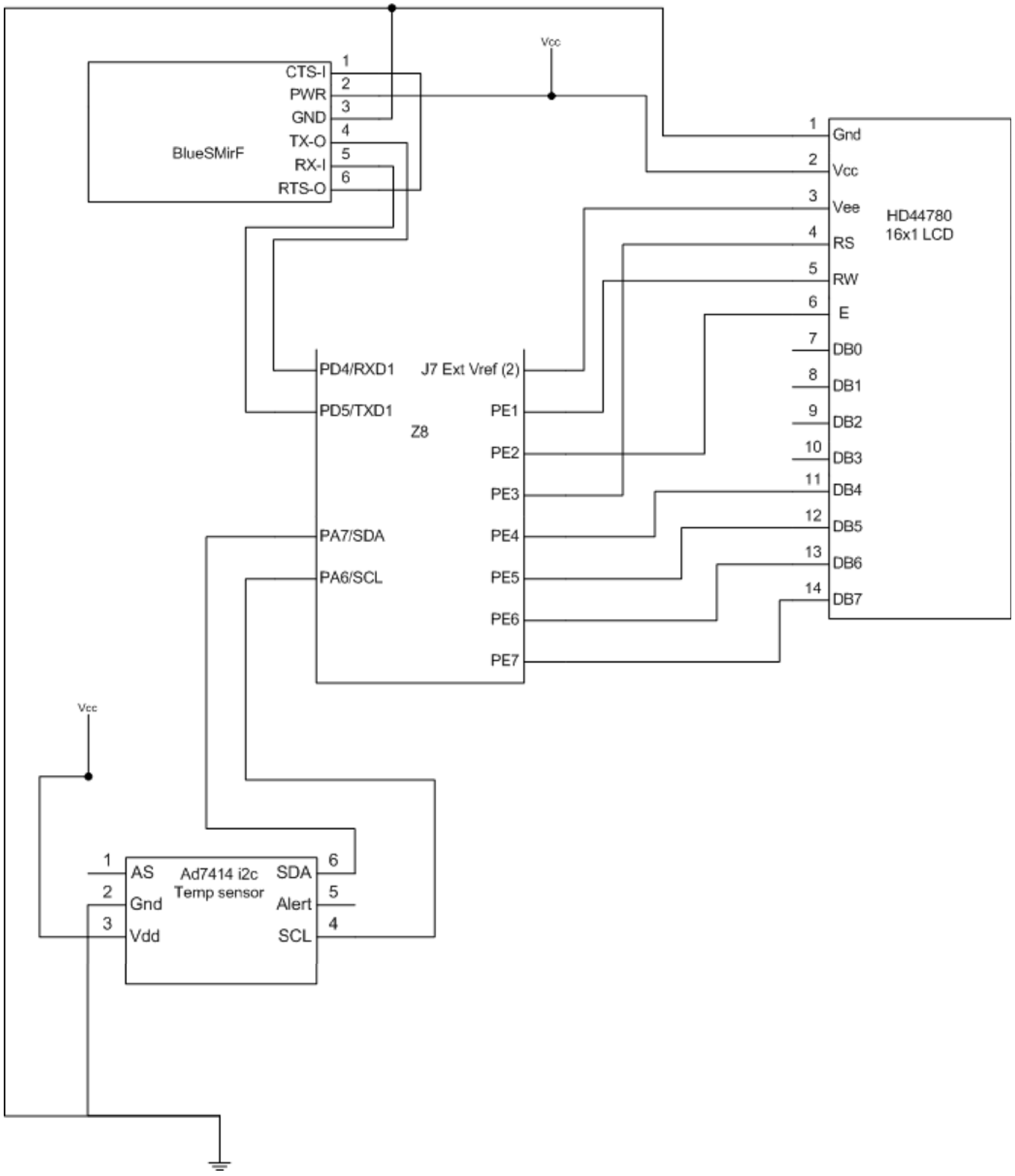
Hardware Interface

Overview



Hardware Wiring Diagram

All external components are directly connected to the Z8F6403 microprocessor.



Pin Connection Detail

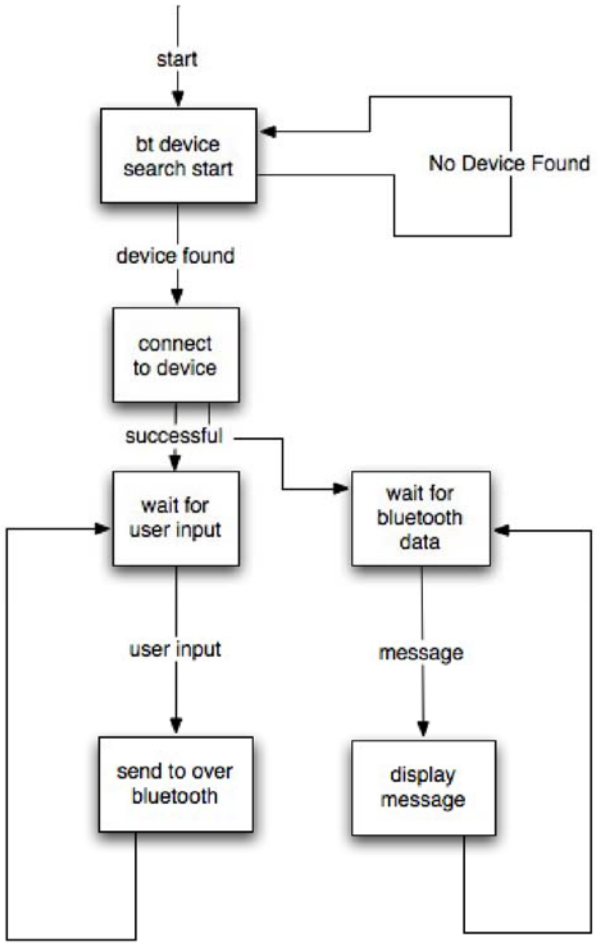
Z8	**LCD**	**Description**	
Gnd	Pin 1	Gnd	Ground
Vcc	Pin 2	Vcc	Power
J7/Ext VREF (2)	Pin 3	Vee	Contrast Control
PE3	Pin 4	RS	Register Select
PE1	Pin 5	RW	Read/Write
PE2	Pin 6	E	Enable
-	Pin 7	DB0	Data Bus
-	Pin 8	DB1	
-	Pin 9	DB2	
-	Pin 10	DB3	
PE4	Pin 11	DB4	
PE5	Pin 12	DB5	
PE6	Pin 13	DB6	
PE7	Pin 14	DB7	

Z8	**BlueSMiRF**		
	Pin 1	CTS-I	Clear to Send
Vcc	Pin 2	PWR	Power
Gnd	Pin 3	GND	Ground
PD4/RXD1	Pin 4	TX-O	Trasnmit
PD5/TXD1	Pin 5	RX-I	Receive
	Pin 6	RTS-O	Ready to Send
*CTS-I is looped to RTS-O to disable hardware flow control.			

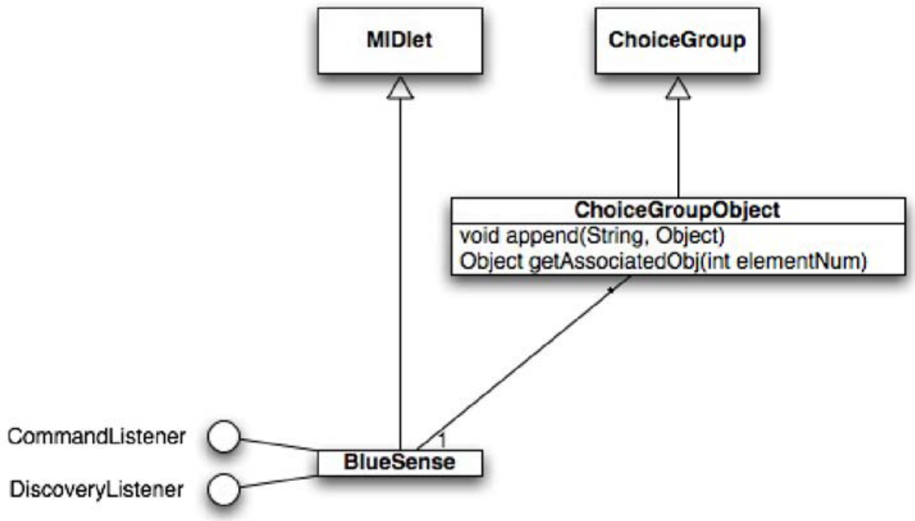
Z8	**Temp Sensor**	**Description**	
-	Pin 1	AS	Address Select
Gnd	Pin 2	GND	Ground
Vdd	Pin 3	Vdd	Power
PA6/SCL	Pin 4	SCL	Serial Clock
-	Pin 5	ALERT	Alert Config
PA7/SDA	Pin 6	SDA	Serial Data

Software (P910i)

P910i client program flow diagram



P910i Client Program Classes



BlueSense Class

This is the main class of the client program. It extends the javax.microedition.midlet.MIDlet as required by J2ME programs. In addition, it implements the following Interfaces:

- CommandListener: allows user input to execute certain commands from the UI.
- DiscoveryListener: bluetooth api for device discovery.

ChoiceGroupObject Class

This class extends the ChoiceGroup class. This allows the client program to keep track of remote devices (or any objects) in addition to the display strings. It is used by the BlueSense Class to display and keep track of discovered bluetooth devices found during the search.

P910i files:

- *BlueSense.java*: source file for main program. Belongs to hp.blue package.
- *ChoiceGroupObject.java*: source file for ChoiceGroup class. Belongs to hp.blue.package.
- *bluesense.jad*: describes the jar file that is loaded onto the P910i.
- *project.properties*: used by WTK to profile the J2ME program.

P910i User Interface

There are two main forms/windows for the client program. The first is a search form and is loaded at startup. The second form is the main program and waits for user input and display responses from the bluetooth connection.

The Search Form starts the bluetooth device search function. When a device is found, it is added to the list of choices which is updated dynamically and displayed on the form. The first device found is also highlighted as the default choice. The user may select another device from the list. There are three commands available on this form:

- RESCAN: restart the bluetooth device search. Currently, the device list is not cleared and duplicates are not checked for, so the same device may show up multiple times.
- CONNECT: connect to the highlighted device. If no device is selected (or found), a rescan is performed when this button is pushed.
- EXIT: this command resides in the menu. it will exit the program.

The Main Form has a user input text box and two text fields, sent/response. The text box allows the user to input text using the natively available interface (QWERTY keyboard, touch screen/stylus, or T9 input using keypad). The *sent* text field is immutable and echoes what was sent to the bluetooth connection. The *response* field is also immutable and outputs what is read from the bluetooth connection after data/command has been sent. This form has two commands:

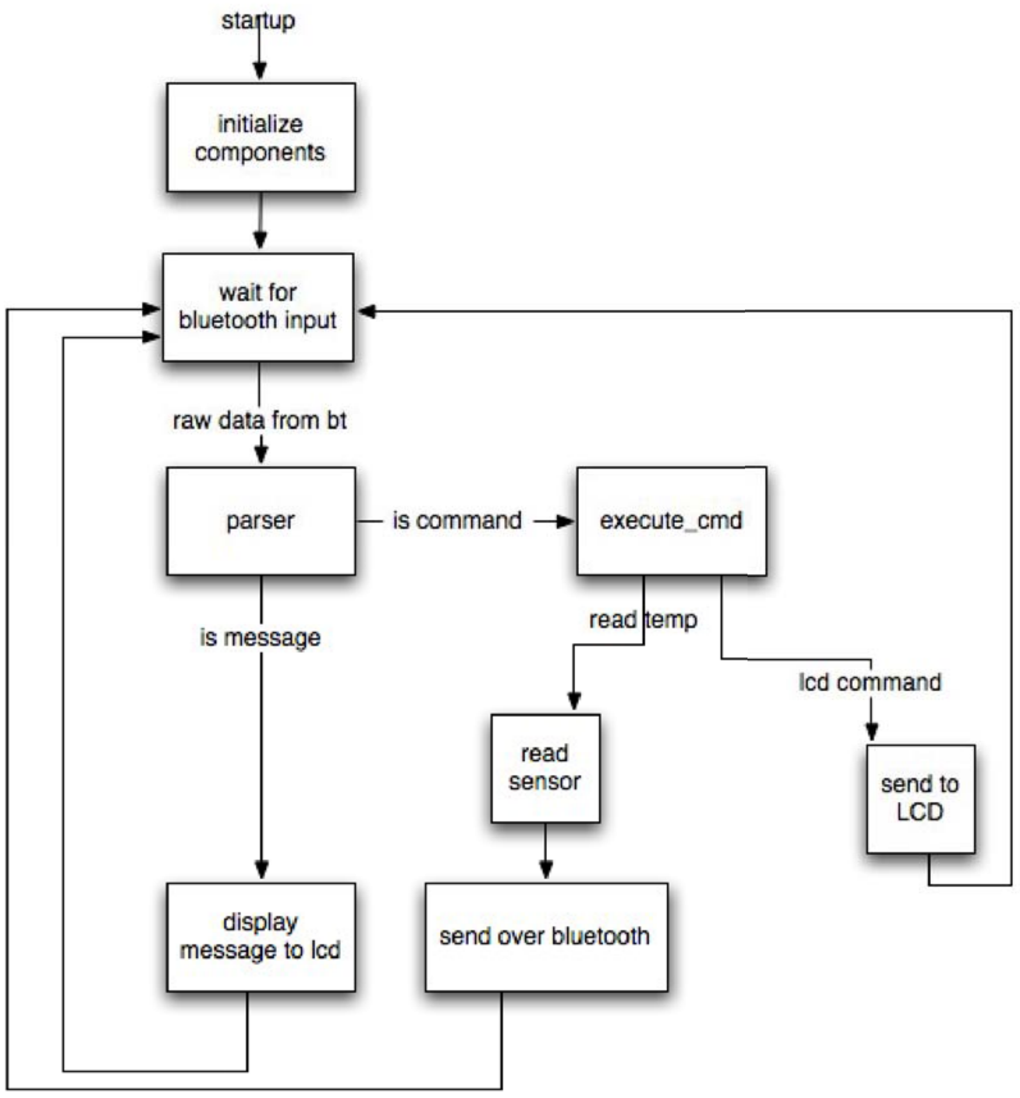
- SEND: sends whatever is in the text box to the bluetooth connection.
- EXIT: resides in the menu. exits the program.

P910i Bluetooth Communications

This client program uses the APIs specified by JSR82 for bluetooth communications in addition to various standard J2ME classes. When a user has selected a device, the program will create a service URL by combining *"btspp://"* with the selected device's bluetooth address followed by *":1"*. This specifies service channel 1 on that device. An example service URL would look like *"btspp://00A09617A712:1"*. A connection is established to this device by using the StreamConnection class on the service URL. Once this happens, an OutputStream and InputStream is created to allow read/write to the remote device. A protocol was created to allow proper read/write termination on the this (the client) end and to prevent the client from blocking on a read. When the client receives a '\r\n' in the data stream, it signifies that the server will not be sending any more responses for the last request. Otherwise, the protocol is a simple request/response loop. Interestingly, I did not have to configure baud rate or other serial settings when pairing with the BlueSMiRF.

Software (Zilog Z8)

Zilog Z8 server program flow control



Z8 program files

- *main.c*: initializes components, program startup.
- *common.c / common.h*: common helper functions, such as sleep.
- *ad7414.c / ad7414.h*: temperature sensor driver
- *i2c.c / i2c.h*: i2c bus driver
- *lcd.c / lcd.h*: lcd driver
- *ui.c / ui.h*: user/machine interface for server. reads input from uart and responds.

Temperature Sensor Driver

Using the provided i2c driver (provided Professor Dan), a simple driver was written for the ad7414 temperature sensor. It has only one function, *ad7414_getTemp()* which returns the current temperature reading.

LCD Driver

Based on Application Note 143 from Zilog, a custom LCD driver was written for the HD44780 display. Several functions were written to allow easy printing to the driver such as *lcd_prints(char *string)* which prints a string/character array to the LCD. In addition, 8 custom characters were created and can be accessed via a '/c#' command.

BlueSMiRF

The BlueSMiRF has the following default settings from the factory:

- 9600 8N1 serial port settings
- "Sparkfun" friendly bluetooth name
- "default" pass key if required.
- Serial Port Profile over Bluetooth

If using the default settings, no real configuration is necessary. All data sent to the UART is pass to the serial port profile over bluetooth and vice versa. When needed however, the configuration mode can be entered by sending "+++\n" in a communications terminal program that is connected to either end of the BlueSMiRF (physical serial port or serial port profile over bluetooth). Once in configuration mode, there are several commands to query and configure the device settings. A few examples include:

● ATSI,2<cr>	# read the friendly device name
● ATSN,z8blue<cr>	# change friendly device name
● ATSW20,236,0,0,1<cr>	# changes default settings to 57600, 8, n, 1
● ATMD<cr>	# go back into data mode

The last command is required to resume data communications. All commands with a few exceptions will reply with the a message and an "OK". For all the commands and full descriptions, consult the datasheet.

User/Machine Interface

The user interface is a simple command line request/response design. It reads and waits for input from the UART 1 port until a carriage return ('\n') is read and then processes what was received. If the data is a certain set of characters then it is processed as a command. Currently, the following commands are supported:

- /t: read the temperature and reply with the temperature value in Fahrenheit degrees. Additionally, it also sends the temperature to the LCD.
- /c#: print # custom character. There are 0-7 valid values for #. Replies with the custom character number that was printed. The reply gets sent to the UART.
- /s#: shift cursor to position #. This value is in hex and is valid for 0-F. Replies with the position that was sent to the UART.
- /p: print all custom characters to the LCD. Sends the text *'done!'* to the UART.

If the data received does not match a command, it is displayed to the LCD. A *'done!'* message is sent to the UART when the message is displayed.

Hardware & Software Tests

The Zilog Z8 program was developed first and verified using a PC w/ bluetooth support. It would be paired to the PC to ensure the program and user tested to verify correct input and output. Each external component was written and tested separately before combined into one program. The temperature sensor was verified by placing my finger against the sensor to increase the temperature reading. The LCD was visibly inspected to ensure the correct data was displayed. The BlueSMiRF was verified by a small program which sends output through its connection. On the other side of this connection was a PC terminal program displaying the output.

After the Z8 program was confirmed to work correctly, the P910i app was developed. While still developing the UI, the Sun wireless tool kit emulator was used to verify various forms and menus were displayed correctly. Once bluetooth was added to the program, this was no longer possible and it was tested against the Z8 itself. Again, verification was done by user testing (visible inspections of input and output responses).

Milestones

2007-03-20: Established a bluetooth connection between the Z8 & PC / bluetooth

2007-03-23: Loaded sample J2ME apps to P910i. Wrote own simple hello world apps

2007-03-27: Temperature sensor driver written

2007-04-10: LCD driver written, Temperature Sensor connected. Z8 program complete

2007-04-13: J2ME UI designed, Rough implementation complete.

2007-04-15: J2ME bluetooth comms completed. P910i client program completed.

Version 2.0

There are many additions that can improve the operation of the client program. First is to add service discovery rather than simply connecting to service ":1". Service Discovery did not currently work properly on the P910i phone, mostly likely due to my limited knowledge of the J2ME bluetooth API.

The user interface could be cleaned up as well. Instead of a simple response text, a log of messages could be kept so that the user can see the previous messages. Additionally, buttons could be added to the main form to allow easier methods of sending repeated commands, such as retrieving sensor readings.

Auto device discovery and connection could be added. When devices get near each other, they could auto pair up and begin communications. The client program would have to run in the background of the phone.

The communications protocol could be improved to be more generic, allowing other devices such as GPS coordinates to be read in via the client program. Additionally sensors could be added. Support for self contained bluetooth sensors could be added.

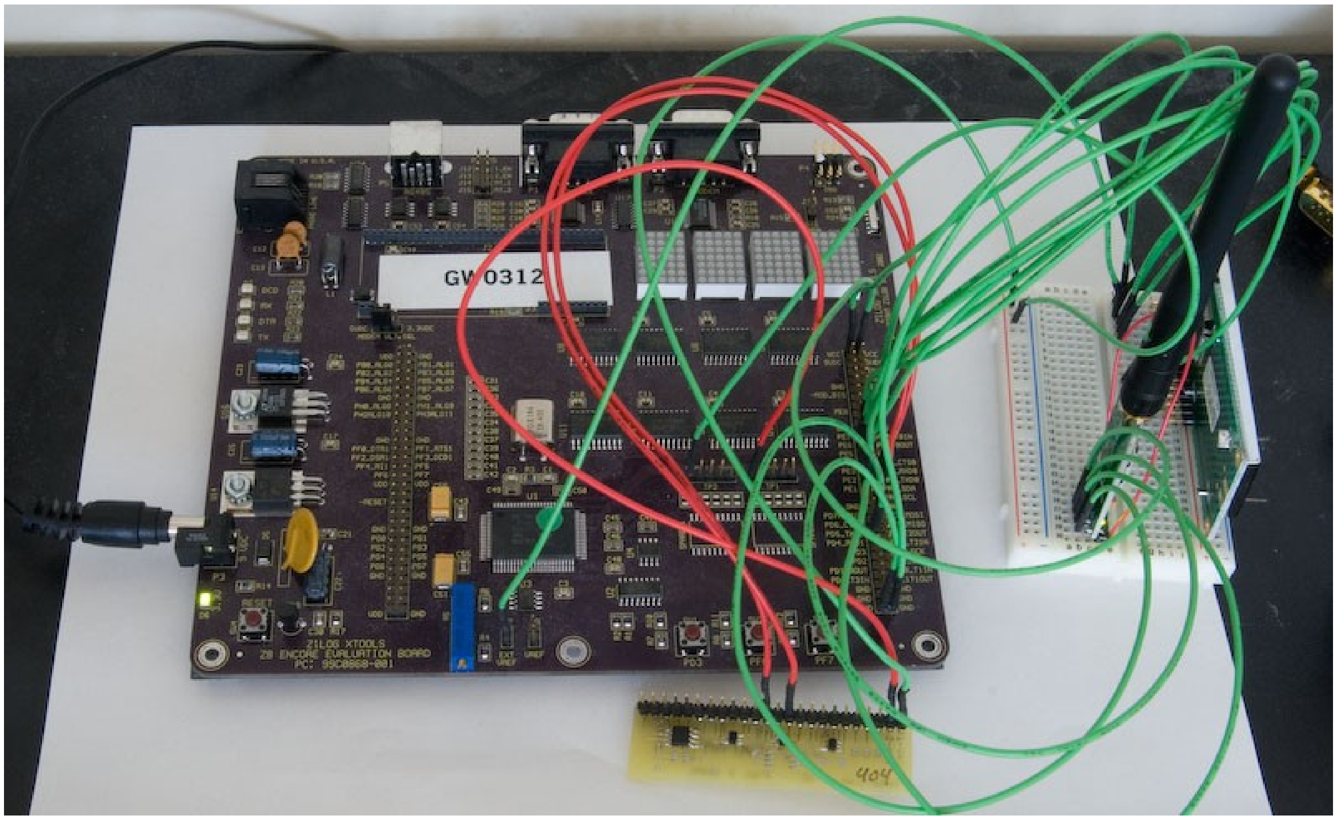
The client could be program to talk to multiple bluetooth devices simultaneously. A system to automate what messages go to what devices would have to be designed, but this would allow the client program to associate and communicate with many devices housing different sensors.

Appendix

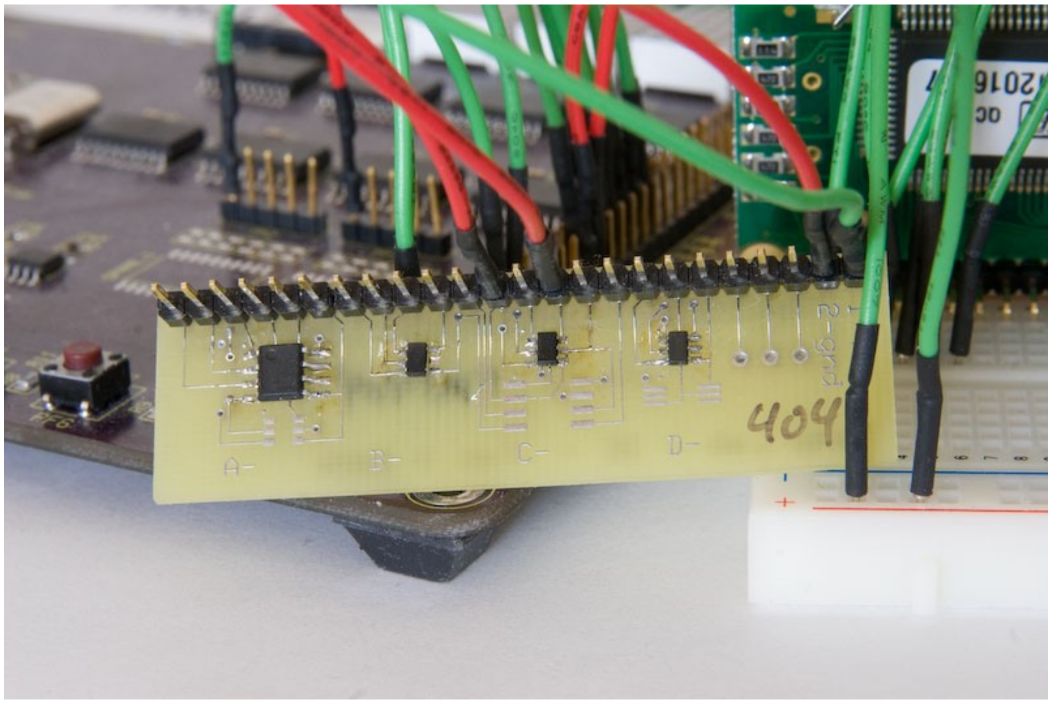
Photos

Sony Ericcson P910i

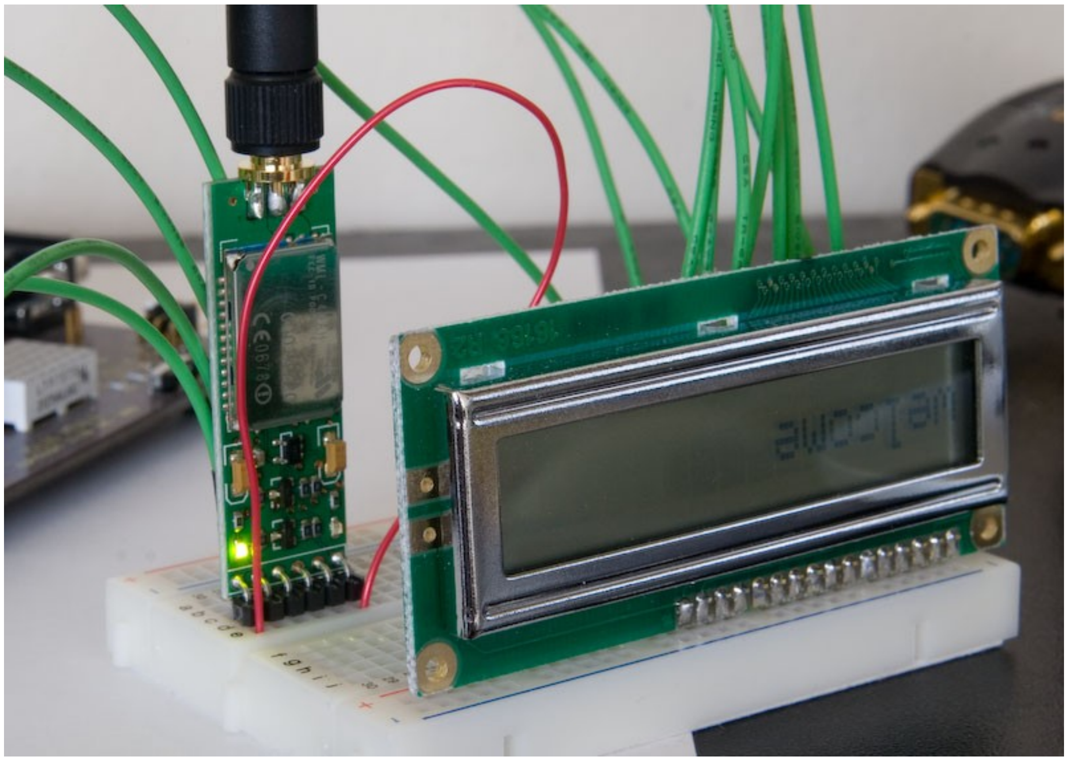
Z8 Encore! development board overview



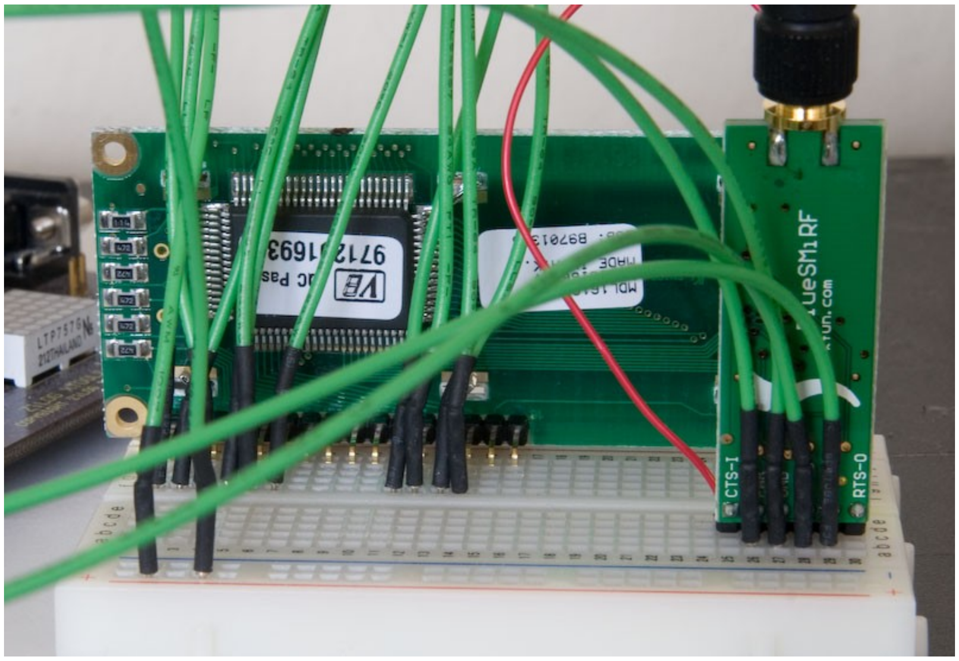
Temperature Sensor



LCD and BlueSMiRF on breadboard

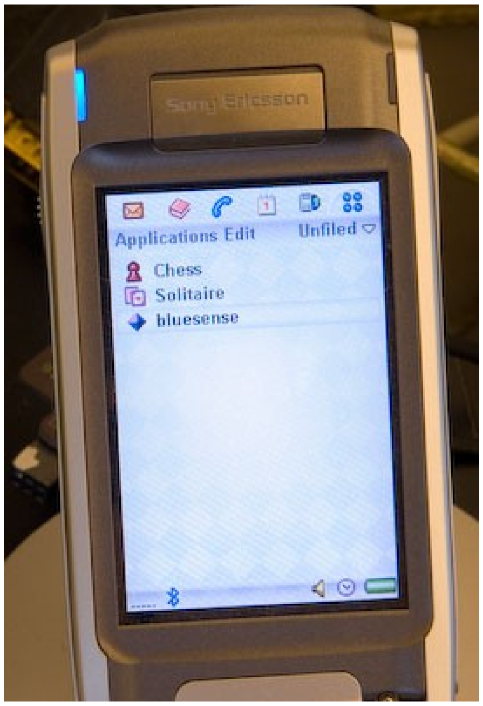


LCD and BlueSMiRF connections (backside)

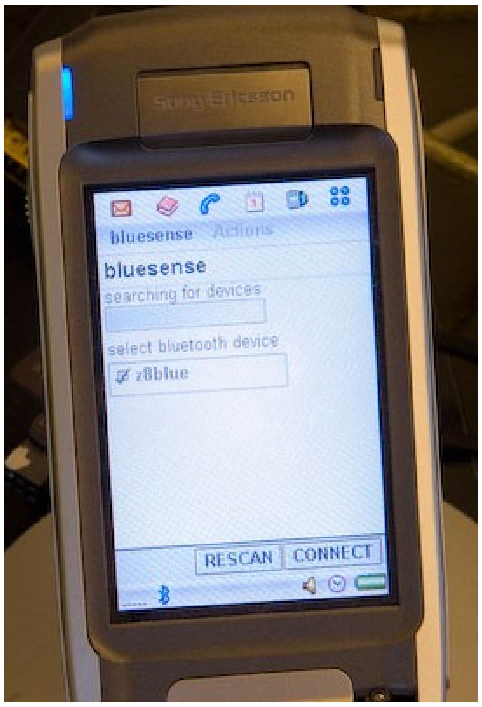


P910i Client Screenshots

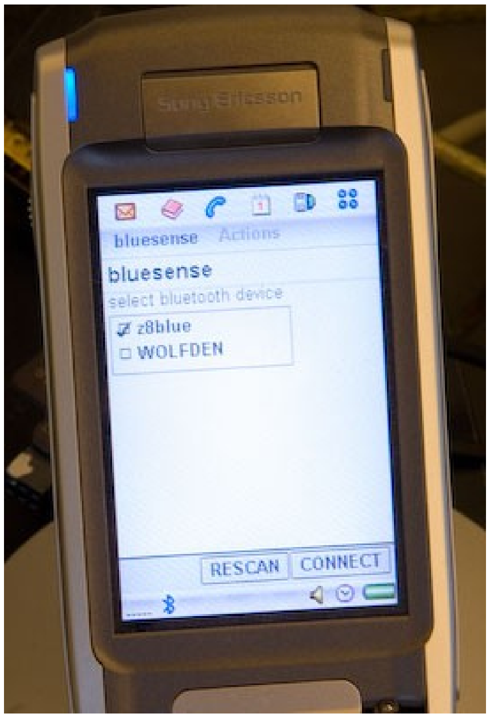
1. Before application launch



2. App startup, device search started (z8blue found)

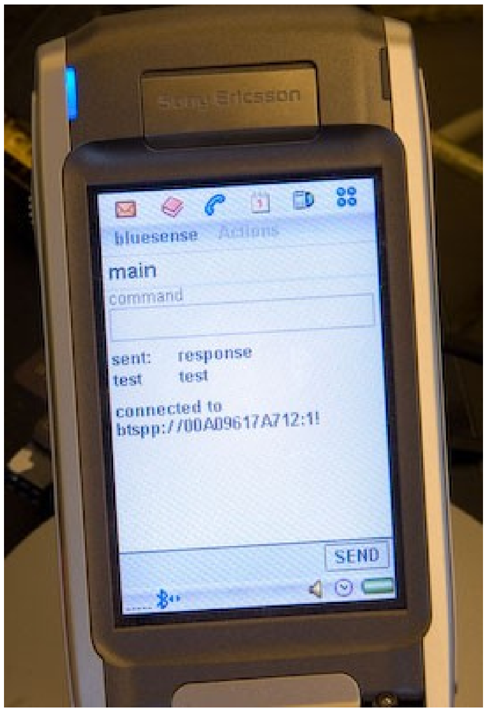


3. Device Discovery Complete (2 devices found)



bluesense
select bluetooth device
☑ z8blue
☐ WOLFDEN

4. Connected to z8blue

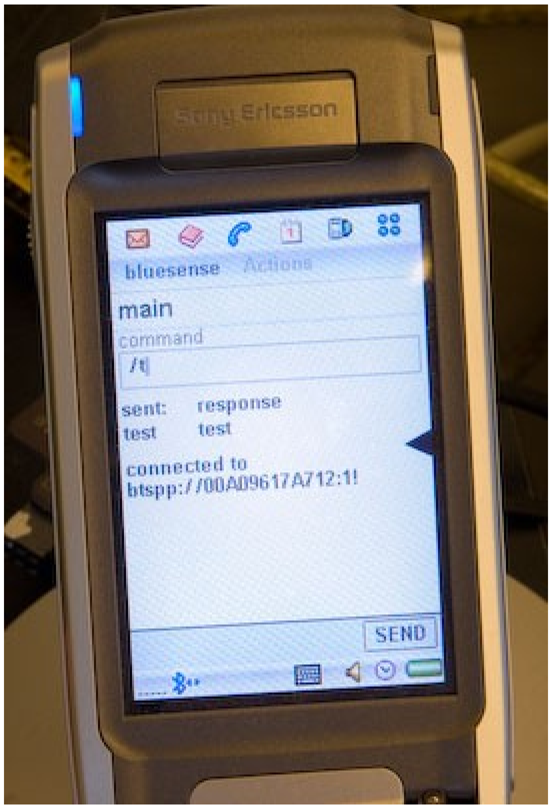


main
command

sent: response
test test

connected to
btspp://00A09617A712:1!

5. User input (/t command, read temp)

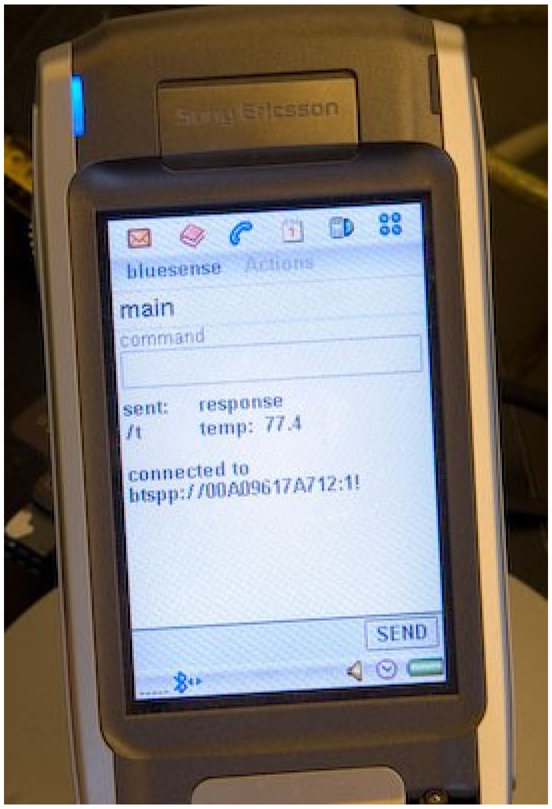


main
command
/t

sent: response
test test

connected to
btspp://00A09617A712:1!

6. Response Received



main
command

sent: response
/t temp: 77.4

connected to
btspp://00A09617A712:1!

Notes

Description of various folders in the zip file:

/bluesense: j2me program for p910i
To compile and run this application, place in the x:\wtk25\apps folder and open within the Sun Wireless ToolKit. "X:\wtk25" should be where the wireless toolkit is installed. Use either bluetooth or usb to load the resulting jar & jad file to the mobile phone.

/bluesmirf: c program for zilog z8
Use the Zilog Z8 IDE to open the projects file to compile & install.

/datasheets: resources and datasheets for this project

- AD7414_7415.pdf: temp sensor
- an0143.pdf: LCD driver application note from Zilog
- LCD-100.pdf: lcd display datasheet
- BlueRadios_ATMP_Commands_Rev_3.5.1.1.0.pdf: command set for bluesmirf
- BlueSMiRF_v1.pdf: bluesmirf user manual
- BlueSMiRF-RPSMA-Schematic.pdf: bluesmirf schematic

References/Resources

J2ME Tutorial, Part 1: Creating MIDlets
http://today.java.net/pub/a/today/2005/02/09/j2me1.html?page=1

J2ME Tutorial, Part 2: User Interfaces with MIDP 2.0
http://today.java.net/pub/a/today/2005/05/03/midletUI.html

Wireless Application Programming with J2ME and Bluetooth
http://developers.sun.com/techtopics/mobility/midp/articles/bluetooth1/
http://developers.sun.com/techtopics/mobility/midp/articles/bluetooth2/

J2ME and JSR82 APIs
http://java.sun.com/javame/reference/apis.jsp

Serial Port Communications
http://www.tigoe.net/pcomp/resources/archives/avr/000749.shtml

http://discussion.forum.nokia.com/forum/showthread.php?t=49736