Web-Enabled Wireless Sensing

F. Naghdy¹, Stephen Davis¹, Gordon Wallace², Peter Innis³,
¹School of Electrical, Computer and Telecommunication Engineering
²Intelligent Polymer Research Centre
University of Wollongong, NSW, 2522, Australia

Abstract
A low cost framework is developed for a wireless web-enabled sensing device. The wireless communication is established through Bluetooth standards. A TINI (Tiny InterNet Interface) board developed by Dallas Semiconductors provides an optimal and very low cost but universal interface to the Internet. This represents the first stage of the work in developing a wireless embedded Internet control system. A review of the previous related work is carried out. An overview of the proposed system is provided and the design and development of each element of the system is explained. The results of the validation carried out on the system performance is presented and a critical analysis of the results is provided.

1. Introduction
Distributed control has always been a superior method to centralised control. Design, implementation, troubleshooting and maintenance of a distributed system require much less effort and cost than when all the sensory information and control loops are processed by one central processor.

The synchronization and integration of a distributed control requires the centralised collection of data. In addition, the operator interfaces to the distributed control system through a supervisory system to issue commands or vary the set points of the system.

In such systems, sensors have traditionally been interfaced to the supervisory computer or the data-logging system through wires. The recent trends have been to interface the sensors and actuator through a network. There are also applications where expensive proprietary wireless communication protocols have been employed to transfer data or receive commands. This work explores how the latest advances in the Web-enabled devices and wireless communication can be employed to make sensing and control in distributed systems more flexible, efficient and cost effective.

Web has traditionally been used to provide structured information and data as either static hypertext pages or dynamic pages driven by a database. The concept of embedded Internet systems or web-enabled devices explores the possibility of providing real time data on Web, or driving a device via the Internet.

In the study conducted in this project a low cost framework is developed for a wireless web-enabled sensing device. The wireless communication is established through Bluetooth standards. A TINI (Tiny InterNet Interface) board developed by Dallas Semiconductors provides an optimal and very low cost but universal interface to the Internet. This represents the first stage of the work in developing a wireless embedded Internet control system.

While the methodology is kept as generic as possible, the work is focused on developing a device which is mounted on the knee sleeve of an Australian Football League player. The device provides information on whether the player has landed correctly after jumping. This helps to keep the player off the sidelines and to reduce the injuries.

The motion of the knee is sensed by an intelligent polymer developed at Intelligent Polymer Research Centre, University of Wollongong. The knee motion is measured by a microcontroller and sent to a Bluetooth device for transmission to a second Bluetooth device. The signal is then passed to TINI and provided in real-time on the Web.

In the course of the paper initially a review of the previous related work will be carried out. The focus will be on web-enabled devices and remote sensing using wireless radio.

An overview of the proposed system will be then provided and the design and development of each element of the system will be explained. In the next part of the paper the results of the validation carried out on the system performance will be presented and a critical analysis of the results will be carried out.

2. Background
The work reported in this project exploits two advances in computer technology and communication to produce a system that is highly flexible in its operation and function. They are short-range wireless radio and web-enabled devices. In this section a review of the previous work related to these two
Wireless sensing and actuation are far more attractive options to wired connections in an industrial or office environment. For example, wireless communication with a rotating or moving machine part is less challenging to implement than when wired. Wireless communication can also create new opportunities for more flexible manufacturing processes by establishing communication between a manufactured component and the production machine to activate the assembly or machining process needed by that component.

The research and development in the area of web-enabled devices is very active. There are new systems and products announced regularly. For example, Philip Research laboratories have recently implemented HAVi (Home Audio/Video interoperability) on the Internet. HAVi is a home networking standard allowing entertainment devices to communicate, cooperate and control [5]. It is developed based on IEEE1394 standard. An HAVi network operates like distributed computing environment. In this prototype application, the entire HAVi Java API can be executed remotely. The client application is connected to HAVi via the Internet. The entire HAVi API is translated by XML [6] and SOAP [7] for transport over the Internet.

μVNC is an embedded module that provides low cost Internet connectivity and interconnectivity of home appliances [8]. This is simply a remote display system that enables an operator to operate another programmable system through network. In the current implementation, a PC is used as the home gateway. Through μVNC a user can choose the code of a favourite TV program from a remote browser viewed on a TV and to request the VCR to record the program.

3. Overview of the System

An overview of the developed system is illustrated in Figure 1. The intelligent polymer sensing the motion of the knee is interfaced to a micro-controller. The data produced by the micro-controller is transferred via a Bluetooth device to another Bluetooth device interfaced to Tini. Tini provides the acquired data on the Web via the Internet. The greatest challenge of the project has been the interfacing of different elements of the system. In the following sections each component of the system will be described and the work carried out to integrate the component in the system will be highlighted.

3.1 Knee Sleeve Sensor

The knee sleeve is made out of nylon LYcra, coated with the intelligent polymer through reactive/oxidative dying process. The electrical resistance of the polymer increases when it is extended and is decreased when it is shrunk. The resistance of the textile can vary from about 1 kΩ to 1 MΩ, although typically it is produce in the 10 kΩ - 100 kΩ.

In order to measure variation in the resistance of the polymer R_{p}, it is interfaced in series to a digital potentiometer with a set resistor of R_{t}. Ideally, R_{t} is set to the resistance of the polymer when unstretched. Now if a constant voltage V_{cc} is applied, the voltage at V_{1}, would be half of V_{cc}. When the polymer...
stretched, the voltage read across the polymer can be determined from:

$$R_p = \frac{R_1(V_{CC} - V_I)}{V_I}$$

3.2 Micro-controller

The analogue signal produced by the digital potentiometer should be digitised, converted to packets and sent to the Bluetooth device. The PIC16c773 microcontroller is chosen for this purpose as it provides all the facilities required. The microcontroller:

- runs at 20 MHs;
- incorporates a 12 bit 6-channel analogue to digital converted;
- provides 21 digital I/O ports;
- offers one UART
- has 4k x 14 words of program memory;
- and 256 x 8 bytes of data memory.

The interface between the PIC micro-controller, and intelligent polymer is illustrated in Figure 2. Since, the Bluetooth kit used in this work has a serial port on it, an RS232 chip is used to convert the voltage for the UART on the PIC micro-controller.

3.3 Bluetooth Device

The Bluetooth device operates based on a protocol stack, similar to a TCP/IP of UDP/IP layered architecture. In this work the Bluetooth device provided in Ericsson evaluation kit has been used.

In this system the Bluetooth device communicates through the Host Controller Interface (HCI). This is achieved by sending appropriate packets to the HCI, as described below. At this stage the Stack software is implemented for the necessary HCI commands for basic functionality. This also happens to be an advantage as it keeps the size of the microcontroller program down to a minimum and saves memory on the controller.

The main packet types used in this software are Event Packets, Command Packets and Asynchronous Connectionless Link (ACL) Packets. The opcode commands and event codes are determined from the specifications PDF file [9]. The opcode in the command packet determines the function to be performed, such as reset, inquiry and create connection. The event code determines which Bluetooth event has occurred such as enquiry complete, connection complete and connection request. These events often contain important information such as connecting Bluetooth’s address and clock offset. The ACL packets are used for transmitting and receiving data and can be as large as 65535 bytes long.

The Bluetooth software consists of five main modules:

(a) HCI Module – It implements the basic HCI functions.
(b) Serial Module - It implements RS232 serial routines.
(c) Bluetooth List Module – It contains a list of found Bluetooth devices from an inquiry.
Bluetooth Device Module – It contains information about the Bluetooth devices itself.

Application Module – It is the main program loop.

The HCI module can only communicate with the serial and applications modules. This prevents the application passing incomplete or invalid packets to the Bluetooth device. With this set up, the application module does not need to know anything about the packet layout. This is all handled by the HCI module by adding the appropriate header to outbound data and stripping headers from inbound data.

The Bluetooth device must first be initialized before any data is sent. This is achieved through the application module calling the appropriate sequence of command from the HCI module. Once initialized, the Bluetooth device can then enter into inquiry mode to search for other Bluetooth devices within range. It then uses the Bluetooth List Module to retain the information retrieved from the enquiry. Once the Bluetooth has a list of nearby devices, it can then proceed to create a connection. The Application Module may send and receive data to the other Bluetooth devices when a connection is established.

At this stage, the stack software implements up to the Link Layer Control and Adaptation Layer Protocol (L2CAP) layer since the other layers in the model are not required. The L2CAP layer is required to transmit data between the Bluetooth devices.

3.4 TINI

TINI is a low cost embedded platform, designed to be interfaced to a wide variety of devices including both home and corporate networks. The first implementation of it was in 1998 as a Java programmable device capable of controlling household electrical goods. TINI has been further developed by Dallas SemiConductors and the TINI SIG (Special Interest Group) and the result is a broad platform including software and hardware that can be used to create intelligent network devices. Targeted devices have a small footprint, low power consumption and are cost sensitive.

The main components of TINI are:

- 10 Base-T Ethernet;
- I/O capabilities;
- Dual Serial Ports;
- Dual 1-Wire net interfaces;
- Dual Controller Area Network (CAN);
- 2-wire synchronous serial bus; and
- The DS80C390 microcontroller running a Java Virtual Machine (JVM).

The TINI board is very small and compact, the same size as a 72-pin SIMM module. A TINI module is illustrated in Figure 3.

![Figure 3 – TINI Rev D, 1 MB RAM, 72 pin SIMM, top and bottom](image)

A TINI should be interfaced to a host PC and the network before any development can be carried on it. The board will be visible on the network after it has seen an IP address. It supports both manually entered IP address and Dynamic Host Configuration Protocol (DHCP).

The TINI is interfaced to Bluetooth through its serial port. The Bluetooth Ericsson evaluation kit comes with an interface software to a host PC written in visual C++. A Java equivalent was initially written for PC. After full commissioning after software, it was implemented on TINI platform just by changing the name of the serial port from COM1 to serial0.

The HTTP server software chosen to run on TINI was TINI HTTP server from Smart Software Consulting. It serves HTML pages, supports multi-threading as well Java servlets. A Java servlet is similar to running an applet on the server side, which is ideal for this project. The servlet allows commands to be received on port 80 from a web browser and allow certain functions to be executed utilizing the TINI hardware and/or other devices attached to it.

Java JDK 1.3.1 has been used to compile the servlet, Bluetooth program and TiniHTTPServer 0.16. Before the program is able to run on TINI, it must be converted to a special format. This format is created from TINIconverter which converts all the java class files into a single TINI file. After the conversion, the program can be uploaded to the board using the File Transfer Protocol (FTP). A program called ANT has been used in this work to automate the whole process.

When the program is uploaded, it can be executed in TINI using TELNET. After execution, the server software listens on port 80 for HTTP request and Bluetooth program starts. When a web browser connects to TINI, a HTML page containing a Java applet is loaded. The applet uses standard HTTP requests to call the servlet and it passes the function it wishes to be executed as a parameter. The servlet
receives the parameter and compares the function name with its own list of known functions to determine whether the function name is valid and then executes it.

3.5 Applet

The Java applet is received on the browser from the TINI and provides a Graphical User Interface (GUI) to it. The applet allows the user to initialize the Bluetooth device connected to the TINI board and to create a connection to the knee sleeve. The user is also able to choose the desired resistance values for the digital potentiometer from a set of drop down lists.

The applet uses an unused TCP port to receive data directly from the servlet. It uses a header line to indicate the type of data sent and can then display it in the appropriate window. For the sampled data, the first indicates which channels need to be updated. Then it reads the number of lines as indicated by the update status and converts each line into an integer. It then calculates the voltage and displays it in the corresponding box. A screen-shot of the applet is illustrated in Figure 4.

4.0 System Performance

The validation and testing of the system were carried out in two stages. In the first step, each element was fully commissioned and tested to ensure that performed as was expected. In the second stage, the overall performance of the integrated system was examined.

At first the data flow rate was checked. In order to increase the throughput of the system, the data is transmitted from the micro-controller Bluetooth to the TINI Bluetooth when there is a change. Hence, in the first instance the voltage at each channel was fixed. The meant that the values were sent in the first packet and the following packets only contained the ACL packet information plus one byte status flag indicating that no channels were required to be updated. The test was run for 30 seconds and then again for 60 seconds. The aim was to determine how close to real-time the display was. The buffer on the TINI was used to observe how many samples did not make it to the screen within the measured time.

The test was then repeated by setting the channel one to a signal that was constantly changing. This meant that the data received contained the status bit as well as two bytes of digital data for channel one. The test was repeated by increasing the number of channels with dynamic data one at a time. This meant that the received data was nine bytes in length plus the ACL packet information.

The results are illustrated in Table 1 for a PIC (microcontroller) delay of 3500 in the DO_A2D function, equivalent to a sampling rate of 10 Hz per channel. According to these result, as the number of channels held constant was decreased (which meant increased packet sizes), the number of samples displayed to the screen was also decreased. However, the number of samples in the buffer was increased. This meant that the program could not process the packets in adequate time before new packets had arrived.

When the PIC delay was set to 1000, and after about ten seconds, the serial buffer on the TINI board overflowed and generated many exception errors. The TINI application was then ported to the PC such that the Bluetooth was now directly connected to the PC. The results were similar. After thirty seconds, the buffer would overflow and generate errors. Hence, on a much faster processor and with the PIC sampling at a much higher rate (approximately 50 Hz per channel) the packets still could not be processed in adequate time.

![Figure 4 – Screen-shot of the Applet interface](image)

It should be also pointed out that the packets are fetched by an interrupt service routine (ISR) when the data is available at the serial port. This means that no more data is serviced until ISR is complete. Hence, at high throughput rates, the samples are left unprocessed in the buffer. Adopting a different strategy to decode the packets can remove the bottleneck.

The following factors could be also contributing to the observed bottleneck in the decoding of the received data:

(a) The Java code decoding the packets may not be efficient.

(b) Java in general does not execute as fast as C programming language.
The TiniHttpServer program is still under development and may still have inefficient code/bugs.

The firmware of the TINI board is also under development and may suffer from the same problems in firmware as in the last point. Further work will be required to identify the exact problem.

It has also been observed that the applet runs just as well under Windows as it does on Linux. This demonstrates that once the device is connected to a network, any other Java capable computer attached to the network is able to access the hardware and the data.

### Table 1 – System performance

<table>
<thead>
<tr>
<th>Time(s)</th>
<th>Channels held constant</th>
<th>Displayed samples</th>
<th>Samples left in buffer</th>
<th>No. Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>4</td>
<td>1328</td>
<td>48</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>3</td>
<td>2660</td>
<td>52</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>3</td>
<td>1276</td>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>2</td>
<td>2588</td>
<td>124</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
<td>884</td>
<td>554</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>2</td>
<td>2068</td>
<td>632</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>804</td>
<td>520</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>1</td>
<td>1488</td>
<td>528</td>
<td>3</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>828</td>
<td>456</td>
<td>2</td>
</tr>
<tr>
<td>60</td>
<td>0</td>
<td>1544</td>
<td>520</td>
<td>4</td>
</tr>
</tbody>
</table>

### 5.0 Conclusions

The design and development of a low cost framework for a wireless web-enabled sensing device was reported in this paper. The wireless communication is established through Bluetooth standards. A TINI board provides an optimal and very low cost but universal interface to the Internet. This represents the first stage of the work in developing a wireless embedded Internet control system.

The work is conducted in the context of developing a sensing device mounted on the knee sleeve of an Australian Football League player. Such device measures the tension and stress applied to the knee sleeve when the player jumps. Such information can be analysed to determine whether the player has landed correctly, to keep the player off the sidelines and to reduce the injuries.

The motion of the knee is sensed by an intelligent polymer. The knee motion is measured by a microcontroller and sent to a Bluetooth device for transmission to a second Bluetooth device. The signal is then passed to TINI and provided in real-time on the Web. In spite of the focus of the project, the framework developed for wireless sensing is generic and can be used for any application.

The performance of the developed system has been validated thoroughly at both component and system levels. The results have been satisfactory, though it has been realised that the receiving computer cannot process the arrived packets sufficiently fast at frequencies higher than 10 Hz, when all 4 channels of the micro-controller produce fresh data.

### References