A Controller Based Networked Autonomous Weather Server

S. Cambeşteanu, M.A. Panait, and A.M. Morega
Faculty of Electrical Engineering, POLITEHNICA University of Bucharest, Romania
silviu.cambesteanu@googlemail.com, axel_morisson@yahoo.com, amm@iem.pub.ro

Abstract

The paper describes a SBC-based weather server. Its main advantages over traditional architectures are greater flexibility, lower costs, and higher reliability in the field. By using a proper enclosure, the weather server can be operated outdoors in remote locations, and it can provide information on the main weather parameters such as wind direction average and maximum speed, atmospheric pressure momentary and maximal for the day, temperature, illumination, short term forecasts, etc., in the form of an easy to read webpage that also allows for commands to be passed to the weather server. A networked weather server operates in a distributed monitoring system, covering large geographical areas and helping forecasts with local data.

1. Introduction

A weather server consists of several transducers and gauges, aimed at monitoring weather data, organized as an autonomous weather station around a network-enabled controller, capable of reading, interpreting and delivering the results in form of webpage or specially formatted e-mails. This paper discusses the possibility of building a simple and reliable weather server based on the Z-World "Wolf" single board computer [1].

The system we built is based on the Single Board Computer (SBC) concept. More than a controller, a SBC has also various input and output modules and a network card (LAN adapter), supporting the following interfaces: Ethernet, PPP, and PPP over Ethernet. The programming language for the controller is Dynamic C, a feature-rich, PC-based computer programming language that allows for complex programming to be carried out on a PC and then loaded into the SBC's EEPROM *via* a programming cable. The paper discusses the possibility of building a simple and reliable weather server based on the Z-World "Wolf" SBC [1].

2. Project Description

A weather monitoring instruments comprises atmospheric pressure gauges, temperature, light, humidity, and atmospheric turbidity sensors. Commonly, the software that interprets the signals from these sensors will then provide for short term weather forecasts based on corroborative usage of data, and will make accessible these analyses upon request, along with the regular instrument readouts.

The weather station we developed is equipped with a thermometric sensor provided with a thermistor and a fixed resistor, a piezoelectric pressure gauge, an anemometer for wind speed and direction measurements, and a diffuse luminosity sensor. These sensors have output signals in the 0...+5 V range, and can be directly connected to the inputs of the Z-World SBC. The temperature sensor uses the thermistor to build a voltage divider; the temperature is read across the thermistor and multiplied by a tabulated constant (here, the linearization of the time constant is performed by the table method).

This solution has the advantage of simplicity and reduced cost, as compared to a dedicated sensor. However, the linear LM35 dedicated sensor would be better option if higher precision is required. The resistive potential divider setup has the advantage of a greater thermal constant thus rendering it relatively immune to rapid variations due to special conditions, by reading a mean value of the temperature instead. The pressure sensor is a dedicated piezoelectric type absolute pressure sensor. The strongly non-linear output signal, typical for these transducers, is linearised by the program by using a constant table from the product's documentation.

The wind speed sensor uses two-disk slot optical encoders. One of them is for wind speed, having a disk with a single slot and fixed optical detection sensor. The second one consists of two discs, one with two rows of slots, for marking the true North and the other one for counting angular position increments, and the other having a single fixed slot, oriented by the true North. An "and" logic gate provides for the reset signal to a reversible counter when the total angle exceeds 360^0 and will also change the counting direction.

As a starting point in this project we use a simple solar tracker consisting of a two axis mobile mount powered by stepper motors, which actuates a small solar panel, and possibly a few instruments such as a

diffuse radiation sensor. The tracker is included in the autonomous weather station as support for the luminosity sensor and direct insolation measurements – it is intended for usage in a small solar farm.

The tracker is equipped with a special neural network that enables its autonomous operation. However, in normal operation, it is periodically bypassed, and control is entrusted to the SBC controller for raw positioning, then it is restored to the neural network to make finer position adjustments.

Next we analyze the structure of this application: The system is designed around the Z-World BL2600 Wolf controller. A web based application handles the main parameters display like: air temperature, uptime wind speed, wind direction, relative humidity, luminosity, energy converted by the panel etc. The main functions are presented next.

Panel positioning control. The panels are positioned by using the coordinates computed in the program, using the Equation of Time. This equation has the current date and time as input, and gives the Sun's position in the sky at a certain moment, and two angular coordinates as output. The program transforms these coordinates into steping sequences, for each stepper motor on each axis.

Data logging. Data provided by the sensors is acquired periodically and stored in the flash memory, within the controller. Later, it may be downloaded through the controller's web page. Also the data log can be sent by the end of each day, through e-mail.

Error handling. The controller is capable to analyze the difference between the calculated coordinates and the correction made by the neural network. If this difference is large enough (a threshold of 1 to 4 degrees is specified, depending on the motors and gears), the controller overwrites the correction, reports the error via e-mail, and it follows the computed trajectory that matches the sun position at the specified date. Besides, it monitors all other parameters such as the panel temperature, which must be lower than the maximum value specified by the panel spec.

The core component of the system is a BL2600 controller. The BL2600 is a high-performance, C-programmable single-board computer that offers built-in digital and analog I/O combined with Ethernet connectivity in a compact form factor. The BL2600 is ideal for both discrete manufacturing and process-control applications. It is an advanced single-board computer that incorporates a powerful Rabbit 3000 microprocessor operating at 44.2 MHz; 512K static RAM and 512K flash memory standard; 36 digital I/O: 16 protected digital inputs, 4 high-current digital outputs software configurable as sinking or sourcing, and 16 I/O individually software-configurable as inputs or sinking outputs; 12 analog channels: eight 11-bit A/D converter inputs, four 12-bit D/A converter 0-10; V or ±10 V buffered outputs; one RJ-45 Ethernet port compliant with IEEE 802.3 standard for 10/100Base-T Ethernet protocol; three Ethernet status LEDs; three serial ports (2 RS-232 or 1 RS-232 with RTS/CTS, 1 RS-485 or RS-232); two RabbitNet™ expansion ports; battery-backed real-time clock; watchdog supervisor.

The sensors comprise either a resistive temperature sensor or a dedicated LM35 chip with a greater sensitivity and a linear output (Fig. 1), a piezoelectric pressure gauge, a small solar panel, a diffuse luminosity sensor consisting in several photodiodes with suitable filters, lens and opaque shades, an anemometer with wind direction indicator, and a relative humidity capacitive sensor (sensitive and having a linear output).

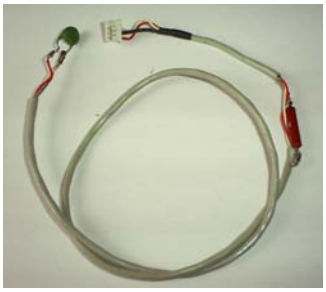

Figure 1. The temperature sensor, based on a 10 Ohm NTC thermistor and a precision resistor. Note that the resistor is near the controller input, and the thermistor is farther away (30 cm of cable) to minimize common mode errors due to the heating of both components.

The solar tracker (Fig. 2) is made of two stepping motors, one two-axis frame, a neural controller and three state buffers for allowing the controller to assume command, fixtures for the solar panel and instruments.

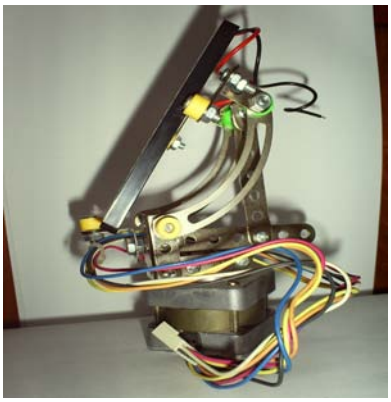

Figure 2.Tracker mount. It is a one axis, adjustable mount for driving a small solar panel (0.25W) used for measurements and tracking efficiency assessment.

The neural network (Fig. 3, 5) simplifies the task of the SBC, as the tracker and the processing unit are entirely controlled by it. Its only input is an "enable" connection, which starts and stops the motor (Fig. 5) driving square-wave pattern generation. The neural

network generates the position control and the fuzzy logic commands under the direct influence of the light gradient sensors.

Figure 3. The signal generator used in testing the neural network, built around an EXAR 8038 precision waveform generator chip and momentarily powered from an external filtered tracking supply (not shown).

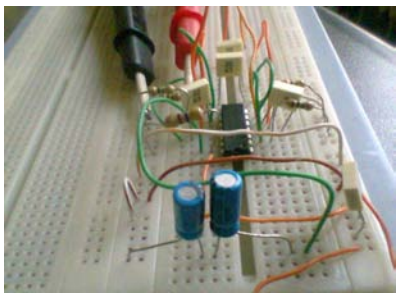

Figure 4. The neural network mounted on a quick-board (no soldering) undergoes tests for stability assessment.

Figure 5. Stepping motor interface (power buffer).

The software. A dedicated application is developed for the reading of time and date, the calculation of the most probable position of the Sun, for recording various parameters such as panel temperature, voltage output, current output, *etc.*, and for either broadcasting over the Internet or displaying them via a web page.

The system's structure is depicted in Fig. 6. The diagram shows the dual-control strategy: the neural network drives the tracker on two axes (X and Y) for orientation and position adjustment, respectively. The SBC pre-positions the tracker according to the *Equation of Time* for the current moment so that the tracking motions are reduced to a minimum. Other sensors connect to the SBC, including the tracker's own encoders that allow the Sun's position in the sky to be calculated and a direct illumination sensor can be read, along with a diffuse illumination one, to characterize the atmospheric turbidity factor.

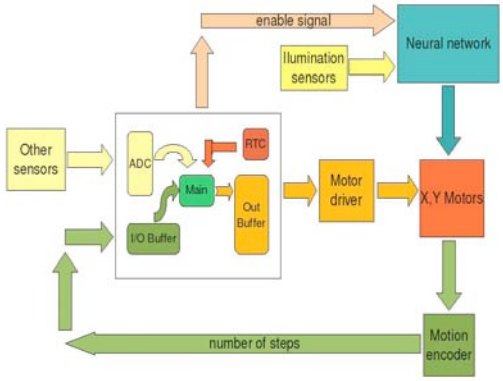

Figure 6. System functional diagram.

BL2600 has a built-in Ethernet LAN adapter, which can be internetworked with other elements. This is a very important feature because data, such as the status of the application or the weather parameters, can be accessed through Internet. The supported interfaces are: a) Ethernet; b) PPP (Point-to-Point Protocol) over a serial link; c) PPP over Ethernet.

The available interfaces depend on the hardware configuration of the target board. If an Ethernet port is available, then it may be used for normal Ethernet or PPP over Ethernet (PPPoE) connection. The TCP/IP protocol is implemented in Dynamic C, which helps developing C language applications for Rabbit controllers. It includes several utility libraries.

The main library is DCRTCP.LIB. As of Dynamic C 7.05, this library is a light wrapper around DNS.LIB, IP.LIB, NET.LIB, TCP.LIB and UDP.LIB. These libraries implement DNS (Domain Name Server), IP, TCP, and UDP (User Datagram Protocol). This, along with the libraries ARP.LIB, ICMP.LIB, IGMP.LIB and PPP.LIB are the transport and network layers of the TCP/IP protocol stack. Dynamic C 7.30 offers support also for a second Ethernet, should such a board becomes available.

To run the TCP/IP stack, the host (controller board) needs to know its unique home IP address for each interface. Interfaces that connect to broadcast networks (*i.e.*, Ethernet) must also have a netmask assigned. The combination of IP address and netmask describes the so-called *subnet*, which is addressable *via* that interface. The subnet basically describes the community of host addresses that can talk directly to this host, without requiring data to pass through a packet router. Point-to-point links need an IP address only, as single hosts.

The network configuration information like IP, MASK, DNS, Gateway, and addresses are stored in the library called "dcrtcp.lib", either static or dynamic: static by using predefined information inside the lib file, and dynamic by enabling the DHCP function (also inside of the lib file).

The program is very flexible: it recognizes

commands like “ifconfig” for setting up the network interface parameters (similar to UNIX), and can also configure the PPP interfaces, usually a slightly more complicated task than for the non-PPPoE Ethernet. The advantage of using PPP is that it can run over a wide variety of physical layer hardware: on Rabbit-based boards it includes the asynchronous serial ports, the Ethernet (PPPoE), and PPP over asynchronous serial allows boards with no Ethernet hardware to communicate using TCP/IP protocols.

PPPoE is often considered a "hack." It seems superfluous to define a protocol that establishes a logical "connection" between two peers on what is otherwise a broadcast (*i.e.*, any-to-any) medium. Nevertheless, the existence of PPPoE was largely dictated by the needs of ISPs who wished to continue using their existing infrastructure, based on the earlier generation of dial-in connections. The availability of high-speed (ADSL, *etc.*) modems, with Ethernet connection to the user's network, makes an attractive alternative out of PPPoE. If the application requires connection to an ISP via an ADSL modem, then most likely, PPPoE support is needed.

PPPoE requires a physical layer negotiation to precede the normal PPP negotiations. This is known as the "access concentrator discovery" phase ("discovery" for short). PPPoE makes a distinction between PPPoE servers and PPPoE clients. However PPP makes no distinction; PPP may be seen as also standing for Peer-to-Peer Protocol. The PPPoE server is known as the access concentrator. The Dynamic C TCP/IP libraries do not support acting as the access concentrator; only the PPPoE client mode is supported. This is the most common case, since the DSL modem is always configured as an access concentrator.

The controller also supports web applications such as HTTP server, hosting for sites or FTP server. Most users of the application will be familiar with at least one web browser (*e.g.*, Mozilla, Internet Explorer, Opera, Safari), and they can use the web site as a graphical interface to operate the controller. In our case we use the controller to provide weather information over the Internet. Then, information may be anonymously accessed. By utilizing the controller any format and layout of the web pages that will be presented to the browser can be implemented. Should the application return information only, without allowing for updates (such as a *data logger*) then forms may be used. Forms, in web parlance, allow the browser's user to fill in some information then submit it to the server. The server, on the application, then performs the requested actions and sends a confirmation back to the browser. This is the most common means for implementing control of the server as opposed to merely querying it.

3. Results

3.1. The WebServer architecture

The server is responsible for fielding requests from the outside world. Each request is analyzed to determine the resource that is requested, the user who makes the request, and whether the user is authorized to obtain that resource.

Table 1. The software routine for a webpage displaying data read from the weather server’s instruments.

```
      form = sspec_addform("myform.html", myform, 5, SERVER_HTTP);

      // Set the title of the form
      sspec_setformtitle(form, "Tracker status");

  // Add the second variable, and set it up with the form
      var = sspec_addvariable("tempNow", &tempNow, INT16, "%d", SERVER_HTTP);
      var = sspec_addfv(form, var);
      sspec_setfvname(form, var, "Current Temp");
      sspec_setfvdesc(form, var, "Current temperature in &deg;F");
      sspec_setfvlen(form, var, 5);
      sspec_setfvreadonly(form, var, 1);

  // Add the second variable, and set it up with the form
      var = sspec_addvariable("windSpeed", &windSpeed, INT16, "%d", SERVER_HTTP);
      var = sspec_addfv(form, var);
      sspec_setfvname(form, var, "Wind Speed");
      sspec_setfvdesc(form, var, "Current Wind Speed");
      sspec_setfvlen(form, var, 5);
      sspec_setfvreadonly(form, var, 1);

  // Add the second variable, and set it up with the form
      var = sspec_addvariable("windDirection", &windDirection, INT16, "%d", SERVER_HTTP
      var = sspec_addfv(form, var);
      sspec_setfvname(form, var, "Wind Direction");
      sspec_setfvdesc(form, var, "Current ind direction");
      sspec_setfvlen(form, var, 5);
      sspec_setfvreadonly(form, var, 1);

      // Create aliases for this form.  This allows the form to be accessed from
      // other locations.
  sspec_aliasspec(form, "index.html");
  sspec_aliasspec(form, "/");

   sock_init();
   http_init();
   tcp_reserveport(80);

   while (1) {
      http_handler();
   }
}
```

If the resource is available, the user is known and has the proper permissions then the resource is made available *via* the browser. A software routine for exporting the data is presented in Table 1.

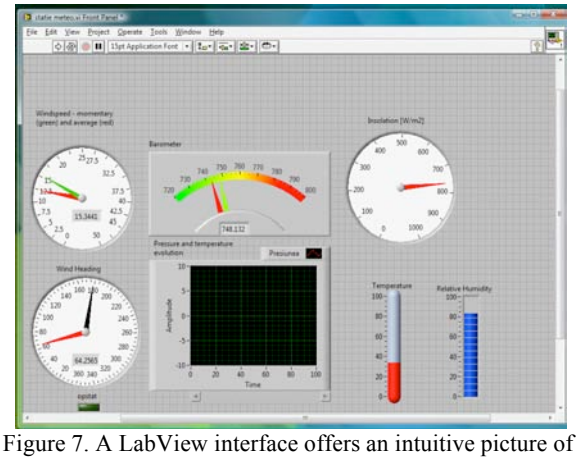

Figure 7. A LabView interface offers an intuitive picture of the measured weather variables.

The user interface/input module can be hosted, for instance, on a LabView server that allows for a simple yet effective GUI (Fig. 7). The interface displays the main instrument's readings, and can support user commands to query or calibrate the system.

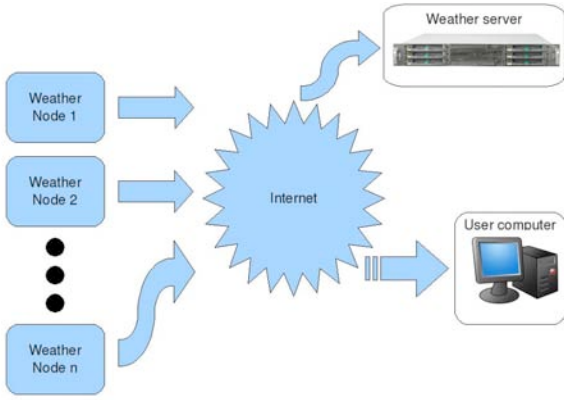

Figure 8. Networked array of webservers.

Finally, Fig. 8 shows a networked array of peer-to-peer weather servers that cooperate in providing regional meteo and weather data. A specialized server (to be implemented at iem.pub.ro) would then concentrate and process the acquired data, for storage and further usage.

4. Conclusions

Recent developments in weather monitoring show the paramount importance of such distributed weather stations, connected in a network capable of covering large geographical areas and delivering in-field data, to aid simulations and forecasts. Modern weather forecast systems are based on such distributed weather-monitoring network, including airport weather stations and small private weather stations as nodes.

This paper reports a simple and reliable weather server based on the Z-World "Wolf" single board computer, equipped with transducers and gauges capable of monitoring weather data, organized as an autonomous weather station around a network-enabled controller, capable of reading, interpreting and delivering results in the form of webpage or specially formatted e-mails. The server is conceived as the elemental cell of a distributed internetworked system, aimed at assisting weather monitoring and forecast, or as part of distributed energy generation systems – *e.g.*, photovoltaic, wind, tidal power farms, etc.

Acknowledgements

Panait M. gratefully acknowledges the support offered by the CNCSIS grant TD /2007-2008 and by the IEM Laboratory.

References

[1] BL2600 Wolf *Reference Manual*, Z-World, http://www.zworld.com (accessed in July, 2008).
[2] I.A. Morega, C. Mogos, M.A. Panait, *Introducere in programare pentru controlul integrat*, MatrixRom, 2006.
[3] Wikipedia: *Equation of Time*
[4] R. Panko, "*Business Data Networks and Telecommunications*", 6th Ed, Prentice-Hall, 2007