Writing Embedded Automation Applications on Low Cost ASUS WL-500g Router

Ing. Bogdan D. Irimia*

**Faculty of Automatic Control and Computers, University "Politehnica" Bucharest, 060042, Bucharest, Romania (e-mail: irimiab@gmail.com)*

Abstract: Due to the rapid development of the computers technology, virtually every automatic system has a central processing unit that embeds all the logic for good functioning. In this paper we propose a system with an architecture based on a low cost central processing unit, consisting of a router produced by Asus, which ca run an open Linux-based operating system. We will describe the hardware platform, the extensions needed for interfacing with the process, the firmware versions available with their features and the ways to implement algorithms on the router.

1. INTRODUCTION

The rapid development of the information technology had a great impact over the automation technology by making available hardware and software platforms with greater processing power, better adaptation to the process and also to the environment, lower costs of implementation and operation and more ease in utilization.

There are two main approaches when it comes to choosing the hardware to run the automation algorithm: choosing a hardware targeted to the actual use case, that has already implemented functions for the needed automation and the developer (the automation engineer) only needs to input the process' parameters, or choosing a hardware that has a wider area of applications (e.g. a PC) and the developer has to write the entire software for automatic control. The decision regarding this choice is influenced by various factors, like price, duration of implementation, reliability, costs of operation, efficiency of the algorithm. In the real world though the hardware platforms available for automation aren't so precise divided in generic platforms and specialized platforms. In fact, specialized platforms can be developed based on generic platforms (e.g. process controllers wit PC architecture) and also specialized platforms can be made more generic by extending its application base (e.g. implementing C compilers for that platform). In fact, de development of most of the systems has at its roots basic programming in C.

Because of the common aspect of having basic functions implemented in C, virtually any hardware that has a processor can be used as a controller in an automated system. Of course, the effort of converting some hardware in controllers is much too great to be even taken into account. But because more electronics tend to operate with an OS, and because most of these electronics are based on something similar to UNIX/Linux, new uses for certain common equipments can be found. In this paper we present such a case of a router equipment that can be used as a controller in an automated system. The choice might seem awkward but there are important advantages that will prove it to be viable.

2. THE HARDWARE

2.1 Basic Configuration

The hardware proposed in this paper to be used as a controller is a router produced by ASUS, based on an ARM processor and running a version of Linux. There are two models of this ASUS router tested, the WL-500g and the WL-500gP (Premium). The first has smaller memory, smaller flash and just one USB port. The next two images show the WL-500g and the Premium model, with their cases and WIFI antenna:

Fig. 1. ASUS WL-500g

Fig. 2. ASUS WL-500gP

We will refer further to the Premium version of the router because the two models are very similar and the code written for one of the model runs on the other one. The ASUS WL-500gP has the following hardware specifications (ASUS, 2009; FreeWRT, 2009):

- 5xRJ45 connectors for 10/100BaseT (1xWAN and 4xLAN, but can be configured as 5xLAN)
- 1xWLAN port, Broadcom BCM4306 802.11b/g WLAN controller
- 2xUSB connectors (the WL-500g has only 1xUSB)
- 32 MB RAM available on the WL-500gP (can be extended to 128 MB RAM using a new chip)
- 8 MB of Flash
- MIPS architecture, Broadcom CPU at 266 MHz

2.2 Hardware Extensions

In order to use the router as a controller in an automated system, it has to interface with the system. The interfacing is done through the serial connections of the router – the two USB ports and also the available RS-232 connection on the circuit board (OpenWRT 2009) – or through a TCP/IP connection. Other connections can be explored soldering extensions to the circuit board of the router (for instance, miniPCI connection could be made theoretically, though it hasn’t been tested).

The serial connections are well suited for automation because there are various sensors with serial output that can be connected to the router. A direct connection to the pins on the circuit board can be made for devices that use TTL levels for their serial connection. As for standard serial connection, a USB to serial adapter can be used. More on the USB to serial conversion in the following subsection. Also, the serial connection can be used to output commands to the system. If an older voltage input or current input is needed for the system, and additional circuit can be implemented that receives serial commands with numeric values and converts them in voltage or current.

Modern automation equipment has also a TCP/IP port that can be used to communicate directly with the router. ASUS wl-500gP has 4+1 RJ45 connectors, each of them being accessible from the software. Automation equipments with TCP/IP protocols include cameras, measuring systems (measuring stations for temperature/pressure/humidity and other meteorological parameters, oscilloscopes etc.), motors, hydraulic systems and so on. We tested the router with different cameras connected on TCP/IP (more details in the software section of this article).

2.3 USB to Serial Conversion

In order to convert the USB signal to standard RS232 or RS485 signal, hardware convertors must be used. There are available on the market several conversion chips form USB to RS232/485, the most popular being produced by Future Technology Devices International Inc. (FTDI) and by Prolific Technology Inc. The RS485 is needed especially in industrial environments and when big distances must be covered, but it has the disadvantage of needing additional power. So we studied the USB to RS232 connection, and if needed an RS232 to RS485 converter can be further used, this one having its own power source.

In order to make the serial RS232 connection available to the operating system and to the application layer of the router, the USB to RS232 conversion chip must be identified by the USB controller, signalled to the operating system, and the operating system must install the necessary driver. In Linux there are drivers (modules) for both the PL2303 chip (produced my Prolific) and the FT232x series chips (produced by FTDI).

The circuit with the conversion chip can be easily implemented (FTDI Chip 2009) or can be bought as a ready-made USB-to-Serial adapter.

3. THE SOFTWARE

3.1 Firmware

The firmware that is present on the router out-of-the-box doesn’t allow writing new applications and using all the hardware in the needed way. Therefore a new version of the firmware needs to be written, one that allows connecting to the console of the operating system, uploading and running applications and configuring scripts. The architecture of the ASUS WL-500gP is able to run various open-source operating systems, so the first choice has to be made in this regard. Choosing the operating system that will run on the router has to take into account the requirements of the applications that will run on the router. The basic requirement of any application useful is to have access to the serial ports (USB with USB-to-serial converter). Also, a development environment has to be available for building own applications.

There are a number of operating systems that can be used (OpenWRT, X-WRT, DD-WRT), but our choice has been the firmware developed by Oleg (ASUS WL-500gP Custom Firmware Page, 2009). This firmware includes the latest developments targeted specially to this router and includes drivers and utilities that allow using the router as a full-fledged controller. Other operating systems (firmwares) aren't specifically targeted to this particular model of ASUS, and they may not be as efficient as Oleg's firmware. Some of them don't support the two USB connections, or include features that aren't needed in automation (like bandwidth monitoring or QOS). Oleg's firmware is lightweight, powerful and is maintained constantly.

The source code for Oleg's firmware is made available on his page, as well as the binary files. The source code is needed if the entire firmware needs to be recompiled. This might be the case when, for instance, a new administration web page is needed – for example, if the router runs a control algorithm, parameters can be set in the administration web page, or connection parameters. The binary file with the firmware has the .trx extension and will be used as an update, in the "System Setup"/"Firmware Upgrade" section of the administrative page.

Once the firmware upgrade has been done successful, a Telnet connection (on port 23) is available to the router. Also, using SFTP, file can be uploaded over the same Telnet connection.

3.2 Scripting and Accessing Resources

Once the connection to the console is available, the operating system looks just like a normal Linux. The difference is that the files are written on a read-only memory or on volatile storage (virtual filesystem stored in RAM). If we want to store files that are available after the reboot, these files must be written on the flash memory. With the command "flashfs" we can access this flash and do basic operations. All the files that need to be non-volatile must be stored in the folder "/usr/local", which is mapped to the flash filesystem. The folder "/usr/local/sbin" can store specific scripts that are automatically run at certain events: before booting ("pre-boot"), after booting ("post-boot"), after firewall enabling ("post-firewall"), after auto-mount sequence ("post-mount") or before shutdown ("pre-shutdown"). These scripts are shell scripts (Quigley, 2002), so they are powerful enough for basic event handling and processing (shell scripts have conditional code and loops).

To save the entire folder /usr/local to flash, the following set of three commands must be used: "flashfs save && flashfs commit && flashfs enable". These commands create an image, write it on the flash and enable the flash to be mounted at boot time. If the image is larger that the flash size, an error will be thrown.

To have access to the local resources – in our case serial ports through USB – these must first be installed. A module has to be loaded in order to make the operating system recognize the device. To list installed modules, the command "lsmod" can be used. To install a new module, the command "insmod" can be used. The module itself is a file with the extension ".o". The file has to be compatible with the operating system, so it has to be compiled for that system. In order to compile any C source code for ASUS wl-500gP, the ASUS toolchain (based on uClibc toolchain) has to be used (MacSat, 2006; uClibc, 2008).

When the necessary modules are installed (pl2303.o or ftdi_sio.o, depending on the USB-to-serial converter, and the generic usbserial.o), the device is accessible through the file in "/dev/usb/". The serial devices appear in the folder "tts" and are named with a number that uniquely identifies each device (from 0 to the number of devices of that type). Reading and writing on serial has to be preceded by the command "stty -F /dev/usb/tts/0 speed 19200", where we specify for each device the connection speed (in this example, for device 0 we specify a baud-rate of 19200). Then, reading and writing is done like reading and writing from/to a file.

The best solution for using serial connection on ASUS WL-500gP is to write a C application that connects to the device (opens the corresponding file) and manages the connection. If the algorithm that runs on the router is complex, the best way would be to write a C application which also connects to the devices and manages the communication, sends alerts and writes logs. For alerting, it is possible to connect a serial modem to one of the USB ports. The modem can be used to send Short Messages (SMS) with the alert or to make a dial-up connection and access services on the Internet (dial-up is made using ppp-dial). If there are not sufficient USB ports, a USB-hub can be used, but the maximum power rating for each USB port must be carefully taken into account.

4. USES AND EXAMPLES

4.1 Common Architecture as a Control Unit

To use the ASUS WL-500gP as a controller in an automated system, the following common architecture can be used:

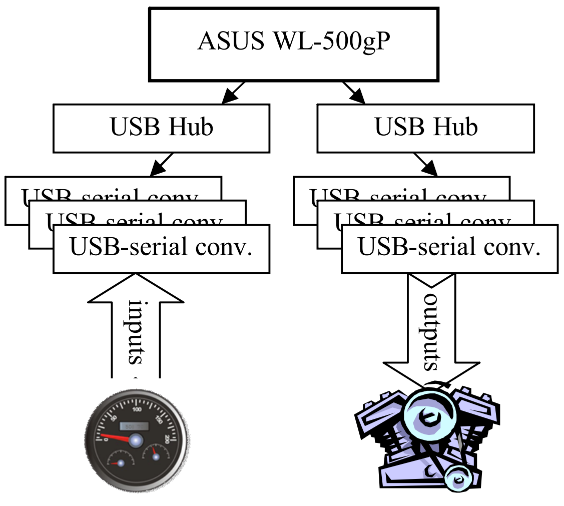

Fig. 3. Common architecture as a system controller

The basic concept is to connect all the devices with a serial connection, for multiple inputs and outputs an USB Hub being used. If the power rating of the USB-to-serial convertor is bigger than the corresponding fraction of the USB's power rating, a separate power source can be used for these converters. The USB hub divides the power of the USB port to each child USB port.

The software structure is based on a single C application (or C++) that implements the following functions: device management (reading/writing from/to serial devices), algorithm logic, settings, logging, alerts (if necessary). The device drivers will be installed at post-boot, using the "post-boot" script file in "/usr/local/sbin". In the same file, after driver installation, the main C/C++ application will be launched. The post-boot script can implement watchdog functions that restart the application if it appears to be frozen. Second level alerting and logging can be also implemented in this script.

4.2 System Supervision

Based on the common architecture presented in the previously, a supervising structure can be easily implemented. The main difference would be that the outputs of the controller aren't commands for the system, but only alerts (on various channels). Thus we replace the output serial connections with a single connection for a GSM (or CDMA) modem. The modem can be used for dial-up or for sending SMS. A second serial connection will be left available for log download and configuration.

The GSM/CDMA modem will be recognized by the system as a serial connection. There are a few modems that include their own USB connection and identify themselves differently. For these more advanced devices, a separate driver will be needed. Also, for modems an ACM driver might be needed (depending on the model). For short messaging, common AT commands can be issued ("AT+CMGS"). But more efficient would be to use a dial-up connection and send HTTP requests to a central server. The server will then continue the task of alerting and logging. To make a dial-up connection, the "pppd" application will be used. The Oleg firmware includes this tool, allowing using GSM and CDMA modems for PPP dial (Newbiefan, 2008).

The second serial connection can be used for on-site servicing and log download. These functions must be implemented by a daemon (a C application) that manages the connection for this port. The application should implement a simple command set. This basic set should include: a command for listing log files, a command for downloading a log file and, if possible, a command for executing OS console commands and returning the output on the serial (a serial console).

When using the router as a supervision system, the watchdog and other failsafe procedures are also needed. These procedures can be implemented as shell scripts and launched from "post-boot" as well.

5. CONCLUSIONS

The ASUS WL-500gP router proves to be a good choice for using as a central processing unit (controller) in an automatic system. Running an embedded Linux, the software platform is powerful and easy to use. The hardware itself offers good performance for processing and enough connection options for interfacing with various devices. Also, the cost of development is low and the development time is short (basic Linux knowledge and programming skills are needed). Even though a router would seem to be an uninspired choice for using in automation, we tried to show that there is solid argumentation to sustain otherwise.

REFERENCES

ASUS, ASUSTeK Computer Inc., viewed 12 February 2009, <http://www.asus.com/products.aspx?modelmenu=2&model=1121&l1=12&l2=43&l3=0&l4=0>

ASUS WL-500gP Custom Firmware Page 2009, Oleg, viewed 12 February 2009, <http://oleg.wl500g.info/>

FreeWRT, The FreeWRT Project, viewed 12 February 2009, <http://www.freewrt.org/trac/wiki/Documentation/Hardware/AsusWL500GP>

FTDI Chip 2009, Future Technology Devices International Ltd., viewed 12 February 2009, <http://www.ftdichip.com/Documents/Schematics.htm>

MacSat 2006, MacSat, viewed 12 February 2009, <http://www.macsat.com/macsat/content/view/30/29/>

Newbiefan 2008, 'GPRS/3G/HSDPA/UMTS USB modem Huawei E220 on Asus router - HowTo', discussion forum, AsusForum.net, viewed 13 February 2009, <http://wl500g.info/showthread.php?t=13480 >

OpenWRT 2009, 'ASUS WL-500g Premium', wiki article, 6 February 2009, viewed 12 February 2009, <http://wiki.openwrt.org/OpenWrtDocs/Hardware/Asus/WL500GP>

Quigley, E. (2002). *UNIX Shells by Example*. Prentice Hall PTR, Upper Saddle River, NJ 07458

uClibc 2008, Erik Andersen, viewed 12 February 2009, <http://www.uclibc.org/toolchains.html>