Thesis Overview:

Protocols for Microcontrollers Networks

Ricardo Antonio López
Facultad de Informática
Universidad Nacional de La Plata
September 2010
lopez.ricardoa@gmail.com

The microcontrollers are embedded in our way of life and are found in many devices. The ability of Very High Scale Integration (VLSI) - with almost exponential growth in recent years makes these devices every day contain functions previously unthinkable. Therefore, a grouping of these networked devices creates a very powerful system, which equipped with a standard protocol that allows interconnection to the Internet gives you a virtually unlimited range of scalability.

For these reasons, this thesis studied the implementation of a network of microcontrollers, defining functions equivalent to those with Supervisory Control and Data Acquisition (SCADA) of great scale. Based on the definition we have, we studied among others, the protocols of communications layers to reach two of the most popular implementations used in industrial environments: Rs485 and Ethernet. While the former is much older, it remains still active and has been promoted from creating compatible interfaces, due to Rs485's high immunity to electrical noise that provides a transparent interface over fiber. The second, more modern, already have set a trend because of its ubiquity and extent of benefits.

In different projects in recent years, it has been observed the difficulty that exists in the Remote Telecontrol Units technology of the 90's, where that poorly distributed equipment, required large cabinets from field data resulting in high costs of wiring, for example, producing a system with low scalability and highly prone to failures. It was essential that raw information from remote units converge to a central site rather that these were to requested and (partially) processed in the field. In the late 90's the paradigm shifted to more distributed systems, with some computing power embedded in boards of small size and low cost, which could be distributed on site, all converging on a network and providing the same information but as distributed components. This installation would be very scalable and provides easy replacement to faulty parts. This interesting property generated the motivation to study microcontrollers' networks, including their performance in the state of the art of this discipline, in order to finally arrive to a definition and a "proof of concept."

The network is conceived as interconnected clusters of Control Units (CUs), each making processes that interact with their own input/output (field) and also with a Control Center of higher hierarchy through the network. There are two main tasks that should be solved in the CUs: a) Local processes interaction with the field (in a closed loop), with the highest priority and b) Network-related processes: synchronization (high priority) and SCADA, (low priority). The network-related processes are implemented using a message passing scheme established by the protocol. The protocol follows the master-slave behavior, which operates a scheme equivalent to a protocol client-server in a synchronous mode. In effect, each command issued from the Control Center, has a number and a destination address. This command runs on any of the CUs of the network, as in a server that performs a local process and returns a result with a message to the Control Center.

The interconnection network has some variations in its lower layers: 1) Multipoint bus (Rs485) o 2) Radial (Ethernet). Devices are interconnected using two types of networks, with the following components:

1. Microcontroller chips, called Control Units (CU1, CU2... CUn), each interacting with a portion of the field´s input/output and running its own application. Also all CUs are linked to a "master" device centralizing some information.
2. Communications Hub, with its specific application, connected to the CUs outlined above using a multipoint network, called Network 1. In turn, the Hub is connected via another interface to the hardware of the Control Center, forming a network called Ethernet Network 2.
3. Control Center, with a Supervisory Control and Data Acquisition (SCADA) application that runs on a Host (PC), which is linked to the Network 2.

The whole network has all the virtues, attributes, and also the difficulties of a distributed system: lack of global time, lost messages, and other devices failure/s. For these reasons, many of the assumptions and policies to be implemented will be similar to those used in such systems.

The **Control Units** have an application that under a general scheme of control, data acquisition, and communication with the network, can embed the code specific to the particular feature needed to be implemented. This configuration code does not include specific specific design guidelines, but several general ones, in order to allow containment in the embedded system. The CUs have the task of interacting with the field and the number of CUs will depend of the complexity of the application system. CUs basically have the following capabilities: a) Acquisition of binary states (Digital Input). b) Acquisition of analog values (Analog Input). c) Acquisition of Sequence of Events (SOE), with precision below 10^{-3} second, and d) Control on the field (Digital Output).

The **Communication Hub** mission is being a bridge or gateway, between the Network 1 (Rs485, interconnecting the CUs), with the Network 2 (Ethernet standard), which can interconnect more hubs and the Control Center. This scheme allows a device configuration tree with the hub at the root. Then, a group of these hubs is linked to one or several Control Centers. Logically, pooling a significant number of CUs through each hub involves an important amount of data flowing virtually from every hub. This situation leads to a higher requirement on the Network and therefore Ethernet has been selected.

The concentration of CUs through a multipoint link, determines the use of a master-slave protocol in order to simplify the access control of the communications bus. This situation determined the Hub to be the master, so that the communications onto the bus are channeled through him, in route to the Network 1. This gives an interesting property of the controlled bus, which can be exploited for other core function of the Hub, which is the **Internal Synchronization** of the CUs in the corresponding network, using a technique of **Embedded Broadcast**. The synchronization performed by each hub over his dependent subnet, allows a highly accurate clock synchronization of each CU. It can achieve also the external synchronization with UTC, adding a GPS device to each hub. The segmentation of the entire network into subnets is beneficial, since the addition of external synchronization to each network segment allows a high degree of reliability against failure in any of the subnets and/or hubs.

In the **Control Center**, the SCADA is responsible for acquiring data, through the communication hubs and deposit them in the database server, so that the Man Machine Interface (MMI) can use the data. To achieve this goal, a high-level language is used to fill in the necessary features to ensure its portability. The Control Center itself is another process that usually runs on the same host of the MMI and is characterized as an application that contains the user interface, operating on data supplied by the SCADA. Going into detail, we can say that in some cases these data are supplied through a direct communication between processes (IPC: Inter-Process communication), and in other cases acquired directly from the database.

Based on the topology described and according to the selection of network types to implement, it was also necessary to take many decisions about the communications protocols to implement on the Network 1. In the case of standard Ethernet at the physical layer, TCP/IP is adopted for the upper layers of communications. This is due to the advantages of this protocol, specifically its ubiquity and availability. Then, there was the possibility of defining an application layer adapted to the needs of a specific application.

In case of Rs485 at the physical layer, it was decided to make a full definition of a protocol, taking into account the following aspects: a) To have a completely non-commercial open definition and, also, expandable. b) Rs485 is one of the most reliable physical networks with a good range (over 1000 meters), bus and balanced, making it highly immune to noise with very low cost and high availability. c) The use of this physic standard is very simple from the availability of hardware which can be connected to the UART of the CUs. Is implemented with a physical bus and we can implement a master–slave protocol in a simple manner, getting an almost direct implementation of a broadcast synchronization for embedded systems. Based on the above, we defined a master-slave protocol for multi-drop network, which has been denominated **Mara-1**.

Due to the fact that almost the entire universe of microcontrollers from different manufacturers have the capacity of Rs232 serial communication and bearing in mind that this paradigm does not enable multipoint communications, in the links of the UCs with the Control Center, we chose to convert to the Rs485 standard, that supports unicast directioning of the CUs of the network. CUs keep a slave status, so that through a master-slave protocol and Half-Duplex, report to the SCADA of the Control Center as the master, using the Hub as a bridge between the two networks. Under these conditions, through a balanced bipolar connection, the Master issues a message that all slaves listen and decode in order to know if the message is for any of them. Verified the matching destination address with the that of each CU, as well as the integrity of the message, the corresponding CU transfers it to its upper layers of software for processing, while the rest of the CUs of the network, discard the message staying in listening mode.

The master-slave protocol that has been designed has only one master and multiple slaves. On that basis, there must be synchronization between queries and responses, as in all master-slave protocol, because in a network of this kind exist eventually multiple slaves to query. In this environment, the master needs the slaves acquired data periodically and eventually storing them in its memory. As a result, through a scheme established by protocol messages, the master makes requirements that trigger processes running on some of the slaves which return a result. In the building of communications protocol we have followed a simplified way, the paradigm of Model Interchange between Open Systems (OSI model) of the ISO, specifically in the definition of tasks in each layer. The application layer is kept constant for any changes to be made in the lower layers of communications. This reasoning, allows greater simplicity in exchanging lower layers, depending on the specific usage.

In the case of Ethernet physical layer it saw as natural implementing TCP/IP in the intermediate layers. This combination is standard in embedded systems. In the case of Rs485 physical layer, it was as reasonable to implement the basic model in three layers. The communications model in three layers defines, from top to bottom, the following: 1. Applications (application layer). 2. Communications (transport layer), and 3. Computers (physical layer). Under these conditions and following this paradigm, in our system we have: a) An application layer, which has defined the necessary for the state polling, the implementation of field control and the configuration of microcontrollers, and includes issues related to synchronization. b) A transport layer, which defines all the necessary so that the virtual communications between application layers will be done correctly, both in the sequence of messages, as in the control of errors that could occur in the lower layer of the network, and c) A physical layer, which depend on the existing at the hardware level.

The application layer is a software routine that is included on each side of the implementation of the master and the slave. The master issues commands that have a specific meaning and they originate over the slave the execution of a routine in their collaborative process in order to complete the request made.

In this conception, we have differentiated two types of commands consistent with the definitions described here in relation to CUs applications: System and User commands. The system commands are generally considered necessary for consistent performance applications. On the other hand, the user commands are originated in the need to implement certain messages, which have more to do with the specific application, defined by the action of control be exercised by the microcontroller. We can say the system commands are functions or services that may invoke the user's application. The commands that the Control Center can send to the CUs (or possibly the communication hubs) can be subdivided as: a) Commands for control and data acquisition. b) Configuration Commands. c) Test and Diagnostic Commands. d) User Commands.

The Transport layer has the critical role of providing communication services directly to the application processes running on different hosts (Control Center and Remote Units). This role remains the same for any particular transport protocol to be chosen. In this case, the criteria and definitions were similar for both Rs485 which uses the denominated Mara-1 or Ethernet which uses TCP/IP.

In TCP/IP, the solution is relatively simple adopting the standard rules implemented by manufacturers. In Rs485 it was made a full definition. Thus, the transport layer protocol has mechanisms to make reliable the message received and sent from/to the application layer. These mechanisms are: a) Preamble indicator of the beginning of a segment. b) Indication of the length of it. c) Time-Out between byte and byte in the lower transport sub-layer. d) Check-Sum verification in the upper transport sub-layer. e) Verification of sequence number in the upper transport sub-layer. With all these elements as part of the definition we achieve a very secure and stable protocol.

We achieved a specific development of a prototype Hardware-Software that was used for evaluation of the purposes expected, and the tests gave highly satisfactory results. We developed a communications protocol in three layers that are used in the project and will be helpful for future action. The same was used and improved in successive versions and in contemporary enterprises with the development of this thesis.

The internal synchronization deserves special attention, with a precision of tenths of millisecond. This was accomplished using a communications medium with a relatively low speed (9600 bps), leaving still room to improve results. There may be some improvement working with more precise oscillators certified by the manufacturer because the ones used were low cost and generic. External synchronization with the addition of GPS over multiple hubs and several subnets of microcontrollers would be solved leaving the assessment of their merit factors which was not completed for reasons of length. Also it was remained to be resolved and verified some conditions of multi-drop protocol, especially those that involve efficiency requirements over the Rs485 channel. Indeed, there were special situations demand (scan below 50 ms) that yielded disconnections and even blocking of communications.

Some characterization of the network and protocols were obtained from direct verification of results: a) Multipoint network operating at a speed of 9600 bps (is it possible to rise up and possibly even 57600 bps to 115200 bps) threw an error of synchronization in units of the order of 0.3 tenths of millisecond. b) Although no tests were carried out with long cable for the interconnection of UCs, an experimental evaluation was conducted of 4000 consecutive queries with a scan period of 50 ms (20 queries per second) and there was no frame loss requirement or repetitive (Retry) on any of the 4 CUs that were used in the test scenario. c) The Ethernet network at 10Mbps, has a lower effective tax rate of 2Mbps because it uses a synchronous channel (SSP) for communication between the microcontroller and the embedded module that limits the speed. However, this speed is high enough for most cases and the network itself showed a very adequate performance and persistence in the connection, allowing foresee that its use can be extended to the communications between the hub and the CUs. In noisy environments it could be made via optic fiber connections with standard converters with virtually no changes to the application prototype.

About the software for the CUs operation we determined empirically (through simulation), with 20 MHz crystals: a) The main routine in a typical situation is resolved in 250 us. That is, the main loop is accomplished 4000 times per second. b) In terms of making the take of state routine that time is raised to 350 us. c) Interrupt routines can be taken at times of less than 50 us.

Ing. Ricardo A. López
lopez.ricardoa@gmail.com