IEC-60870-5 application layer over TCP/IP for an Open and Flexible Remote Unit

Verónica Medina, Isabel Gómez, David Oviedo, Enrique Dorronzoro, Sergio Martín, Jaime Benjumea, Gemma Sánchez
Departamento de Tecnología Electrónica
Universidad de Sevilla
vmedina@us.es

***Abstract*-This paper presents the development and test of the standard IEC-60870-5 application layer protocol over TCP/IP for a Remote Terminal Unit (RTU) based on open hardware and software. The RTU hardware is an embedded system, a SoC-type design using FPGA that has been programmed with the open core LEON with Linux operating system running over it, so both the hardware and IOS are open source. For prototyping the GR-XC3S-1500 board has been used. There is no open source code available for the IEC standard protocols, so application layer protocol over TCP/IP has to be implemented. All the software design has been made in a PC platform using standard development tools. The source code generated for the protocol has been compiled with the standard Linux gcc compiler in LEON. Several tests have been made to prove that the RTU works correctly.**

I. INTRODUCTION

The first steps in the development of an Open and Flexible Remote Unit applied to telecontrol/telemetry is presented in [1]. The main idea is to develop a Remote Terminal Unit (RTU) that is open both in hardware and software.

The hardware platform is an embedded system, a SoC-type design using FPGA. The FPGA itself has been programmed with an open core called LEON [7], an SPARC compliant system capable of running Linux for SPARC. The processor is an open core (i.e. an open hardware) , this means that hardware platform is open, and also it is the operating system running over it (Linux Debian for Sparc has been installed in the system [8]). So, the RTU is, in essence, a Linux-Sparc system [2]. This means that every program available may be used for this platform and, more important, the whole software can be developed in a very similar way than in any standard Linux programming environment.

For prototyping the RTU, the GR-XC3S-1500 board, Fig. 1, has been used. This board is supported by the co-operation between Gaisler Research and Pender Electronic Design.

For communication, the transmission channels available [1] in the RTU are Radio Frequency (RF), GSM (Global System Mobile) and GPRS (General Packet Radio System).

Telecontrol protocol stack usually implements the specification provided by the International Electrotechnical Commission (IEC), called IEC-60870-5 [3]. This document, which specifies a suite of protocols, is divided into six parts

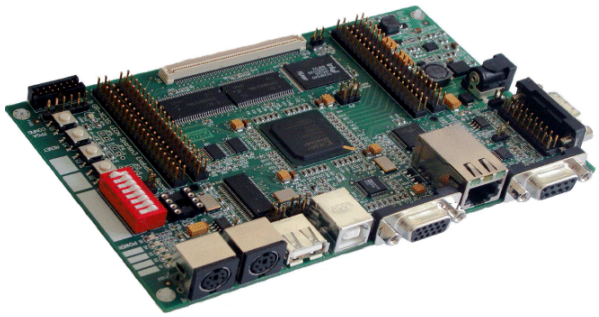


Fig. 1. RTU Prototype.

and specifies an application-layer protocol and a data-link layer protocol. This standard also defines a combination of the application layer using TCP/IP transport services. Although open source software is widely used in Internet, it is not very common in the telecontrol networks area [6], so it is necessary to developed the open software for this protocol stack in the RTU .

The open source implementation and test of the IEC 60870-5 data link layer protocol, described in IEC 60870-5-2, is presented in [4] and [5]. As justified in [1] when the transmission channel is half-duplex, RF, it is necessary to use a protocol that control the medium access, that is what the IEC 60870-5-2 protocol makes.

This paper is a continuation of the RTU software development but focusing this time on the IEC 608070-5 application layer over TCP/IP, that is, Internet. This makes it possible that the RTU could be used in many environments.

This paper is organized as follows; first, an overview of IEC 60870-5 series (section II) is shown. Some details of software design and field testing are described in section III and IV. At the end, some conclusions are presented.

II. IEC 60870-5 SERIES

As mentioned before, telecontrol protocol stack usually implements the specification provided by the International Electrotechnical Commission (IEC) called IEC 60870-5. This series follows the EPA (Enhanced Protocol Architecture) model, which simplifies the ISO standard (OSI model) in three layers, application layer, data-link layer and physical layer. The

standard is divided into six parts and specifies a suite of protocols for both application and data-link layer.

In a typical telecontrol scenario one station (primary station), called CC, controls the communication with other stations (secondary stations), called RTUs, so IEC 60870-5 specification allows real-time telecontrol applications to take place. In this sense, the series defines a set of functions (profiles) that performs standard procedures for telecontrol systems. No all the implementation performs the same functions so an application profile must be described depending on the telecontrol system functionality.

Two different scenarios are possible where or not the CC and RTUs are permanently connected, Fig. 2, or connected via

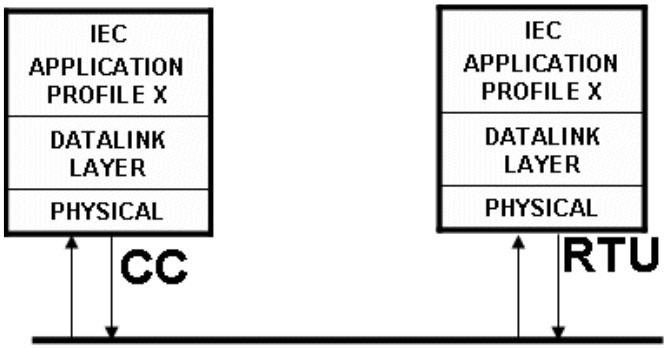


Fig. 2. RTU and CC permanently connected.

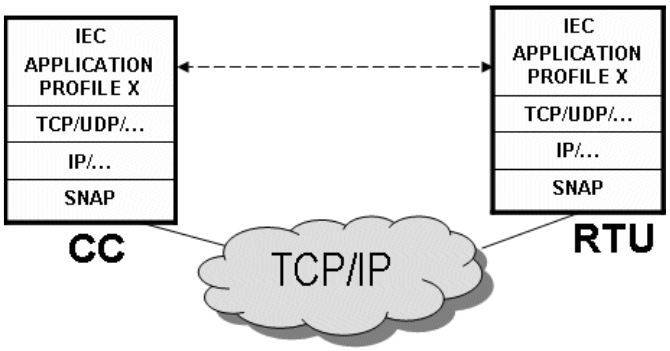


Fig. 3. RTU and CC connected via Internet

an Internet using TCP/IP Architecture, Fig. 3. For the first scenario all the protocols described in the standard has to be implemented, included the IEC 60870-5 (application functions) and for the second one, only the IEC 60870-104 because the other layers are yet implemented on most operating system (TCP/IP stack, the Internet protocols). This paper focuses only on the implementation and tests of the second scenario.

A. IEC 60870-5-5 Overview

This document specifies an assortment of basic application functions for use in telecontrol systems. Each function is composed of transfer procedures of specific ASDUs (Application Service Data Unit) between remotely communicating application processes, that is, the CC application process and the RTU application process.

Selection of application functions from IEC 60870-5-5	Initialization	**User Process**
Selection of ASDUs from IEC 60870-5-104		**Application Layer**
APCI(Application Protocol Control Information)		
Selection of TCP/IP protocol suite		**Transport Layer**
		Network Layer
		Link Layer
		Physical Layer

Fig. 4 . IEC 60870-5-104 . Protocol Structure.

There are application functions to acquire data by polling, to send command, to transmit integrated total, to transmit a file, ... No all the functions must be implemented, so depending on telecontrol application only a set of these functions are available, that is called application profile.

Data acquisition by polling is the only function included in the first profile of the RTU prototype.

B. IEC 60870-5-104 Overview

This document defines the use of an open TCP/IP-interface to a network, containing for example a LAN for telecontrol equipment, which transports data. Fig. 4 shows the general protocol structure, at the RTU and CC, the TCP, IP and PPP protocols are used as transport, network and link layer, respectively. As physical layer, the GSM or GPRS technology are applied at the RTU, and also IEEE 802.3 or 802.11 could be used at the CC.

III. SOFTWARE DESIGN

The software design is divided into two different modules: Control Center module (CC module) and Remote Module, (RTU module). Both modules have been developed in the programming language C++, under the Code::Blocks IDE on a hardware platform x86 (PC) and operating system Linux. Only standard libraries have been used in the development. In the case of Remote Module, the software also has been compiled on LEON (SPARC architecture).

In the Remote Module, two independent processes have been implemented, data acquisition and transmission control. Data acquisition process is in charge of getting the data that have to be sent to CC Module, for test purpose a temperature sensor has been used.

The data saved in memory by the data acquisition process are sent to the CC by the transmission control process. This process behaves as described in IEC 60870-5-104 specification after an initialization phase. The implementation of this process is based on the C++ Sockets Libraries.

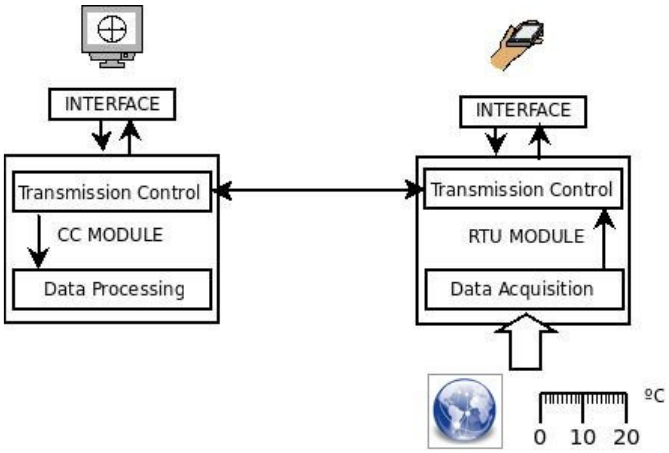


Fig. 5. Software Design.

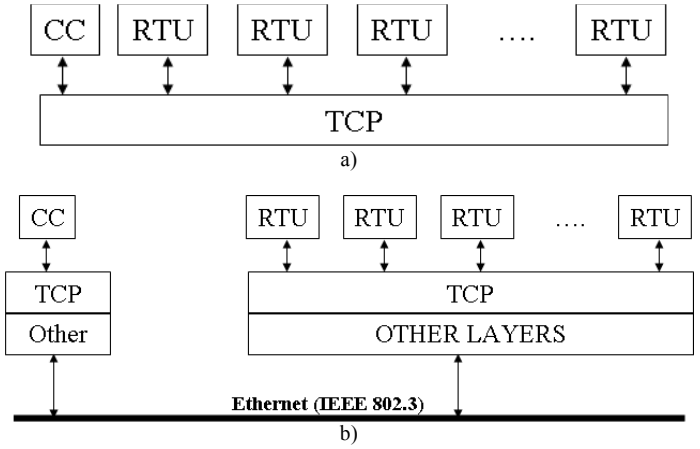
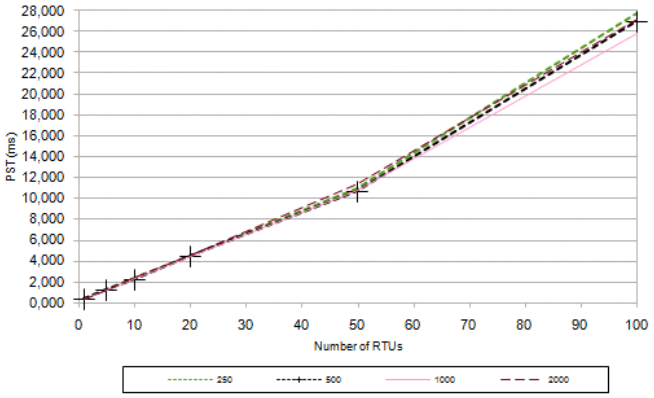


Fig. 6. Test environment.
a) There are n processes that behave like n RTUs and a process that behaves like a CC all executed in a single computer. All processes use TCP services as required in IEC-60870-5-104.
b) There is a process that behaves like a CC in one computer and there are n processes that behave like n RTUs in other computer. All processes use TCP services as required in IEC-60870-5-104.

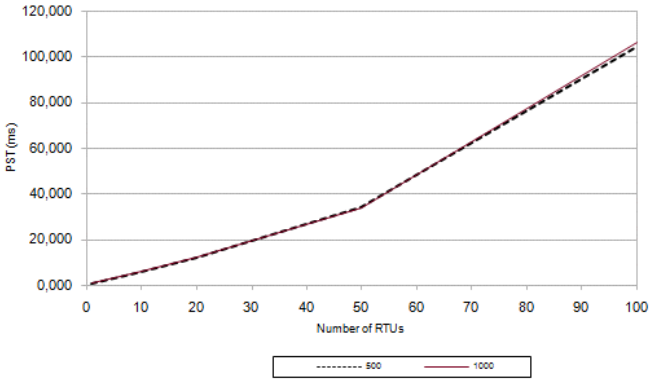
The data acquisition process is replaced by a data processing process in the CC module, which analyzes and prepares the data received from the RTU. The transmission control process in the CC module is in charge of requesting and receiving data from the RTUs.

All functions in the CC and the RTU modules, are available through an interface, that serves as an entry point to the operations asked for by a final user, Fig 5.

As LEON has a standard Linux Debian distribution with standard C++ libraries, getting a binary file is as simple as compiling the same source code developed in the PC. It has been used gcc compiler to compile and link the RTU code in LEON.



a) CC and RTUs in the same computer



b) CC and RTUs in different computer

Fig. 7. PST and Number of RTUs. These two graphics show how the PST changes depending on the number of RTUs, using one, a), or two computers, b), for a total number of 100 polls. The waiting time between a polling of all RTUs and the next polling also has been changed from 250 to 2000 ms in same computer and, only 500 and 1000 in different computers. The differences between tests in one or two computers are due to transmission delays, in this case a Fastethernet link (100Mbps).

IV. Protocol Testing

IEC 60870-5-104 protocol implementation has been successfully tested. One test has been made in order to determine the protocol behavior for real-time application (simulation testing) using PCs and other test has been made to prove that RTU prototype works correctly (field testing).

A. Simulation testing

Fig. 6 shows the simulation test environment. One CC and n RTUs are all executed in one computer (Fig 6.a) or two computers (Fig. 6.b) . As explained before data are acquired by the CC from the RTU by polling. Two parameters have been analyzed, Poll Series time (PST), mean time required to poll all the RTU, and Poll Answer Time (PAT), mean time required to receive an answer from RTUs to a previous poll, both parameters measure how often data from RTU are got.

This simulating test focuses on the number of RTUs and how often each RTU can be polled, or what is the same, the

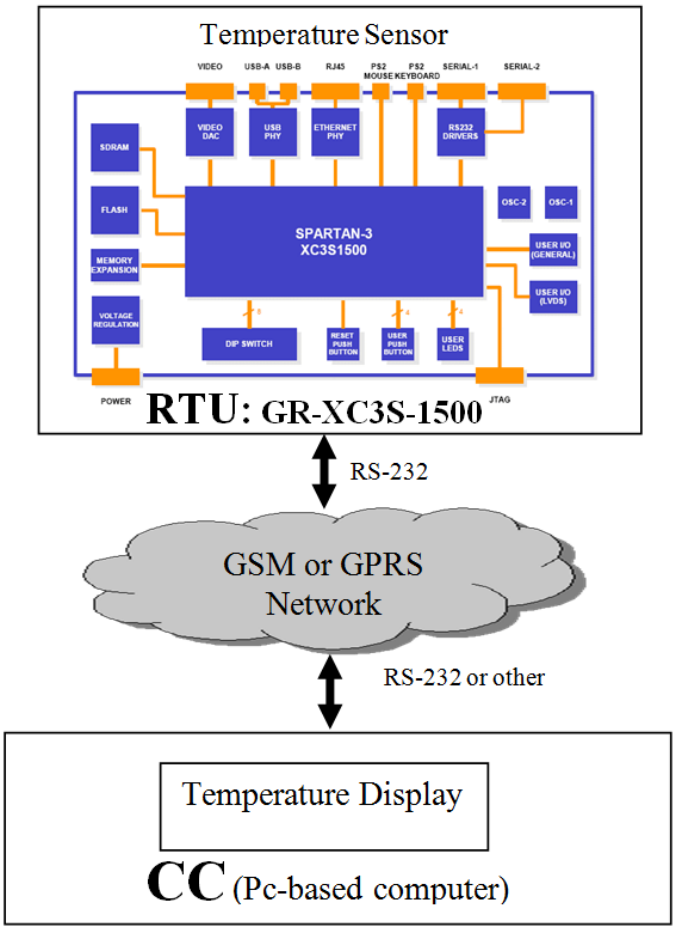
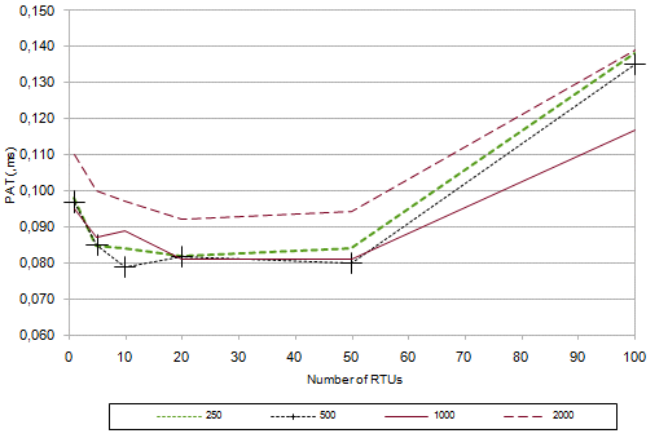
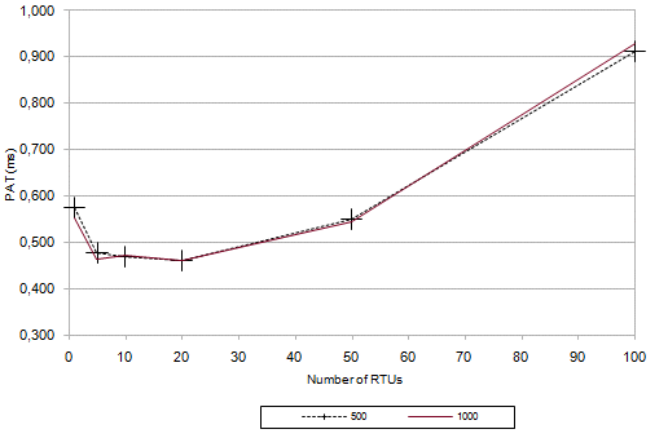


Fig. 9. Field testing scenario.



a) CC and RTUs in same computer



b) CC and RTUs in different computer

Fig. 8. PAT and Number of RTUs. These two graphics show how the PST changes depending on the number of RTUs, using one, a), or two computers, b), for a total number of 100 polls. The waiting time between a polling of all RTUs and the next polling also has been changed from 250 to 2000 ms in same computer and, only 500 and 1000 in different computers. The differences between tests in one or two computers are due to transmission delays, in this case a Fastethernet link (100Mbps).

suitability for real time application. Fig 7 and Fig 8 obviously show how the two parameters, PST and PAT, are incremented lineally as number of RTUs does, but for 100 of RTUs, a very high number of RTUs, in two computers (Fig 8.a and Fig 8.b), the PST is less than 100 ms and PAT is about 0,9 ms, enough for a real time application. In this case the link doesn´t introduce much delay, transmission speed is 100Mbps and transmission delay is almost 0, so the previous claim also depends on the channel characteristics. If the channel operates a lower speed and introduces a higher delay, PST and PAT parameter shall increase their values and maybe not adequate for real time. Next sections shows the field testing using GSM and GPRS channel.

B. Field Testing

RTU prototype, that is LEON, has also been tested only with two of the three transmission channel available in the RTU, GSM and GPRS, Fig 9, because RF doesn´t work with TCP/IP as justified in [1].

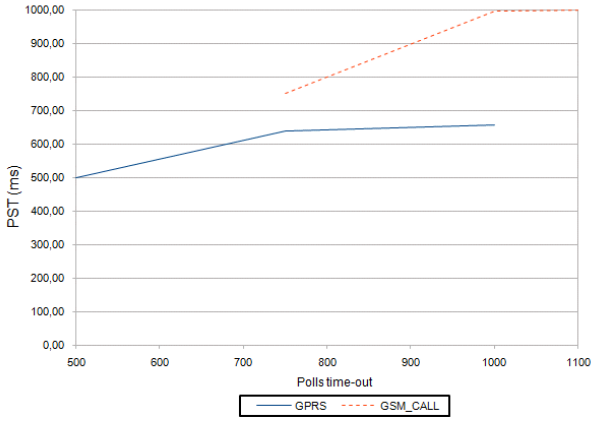
A GSM/GPRS modem has been used, specifically a Wavecom Fastrack M1306B [9]. This modem behaves as a standard AT-command modem via a RS232 port, so the modem is connected to the RTU this way. According to device's specification, it allows a data transmission up to 14.400 bps for GSM and 115.200 bps for GPRS but this feature is dependent on the GSM/GPRS operator used, so it might not be available. GSM operates up to 9600bps and GPRS up to 38.400 bps in the test made.

PAT and PST parameters have been checked for a specific polls time-out, that is, the maximum time the CC waits for the RTU poll answer, Fig 10.a and Fig 10.b. As shown on Fig. 7 and Fig. 8 data processing takes no significant time compared with transmission delays.

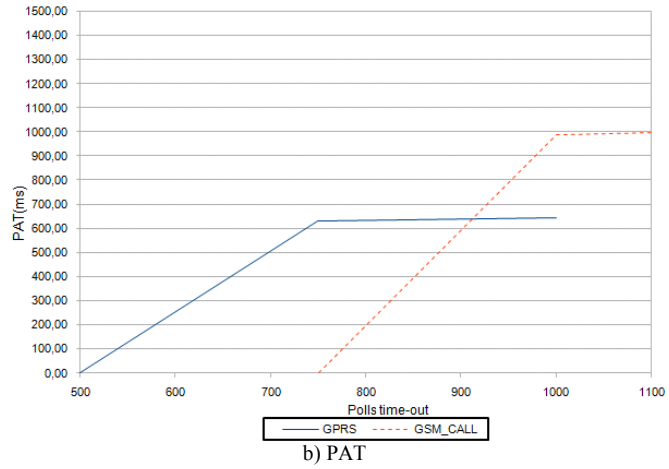
In Fig 10.b answers from the RTU are dropped for time-outs bellow 750 ms for GPRS and bellow 1000 ms for GSM.

Setting a timeout underneath those values implies a high amount of lost replies, but establishing higher time-out values

does not imply higher performance of the protocol. Higher time-outs will generate more successful polls but the polling rate will be decreased. For example, a time-out value of 2000 ms will ensure enough transmission time but as result of this it



a) PST



b) PAT

Fig. 10. Polls Time-out in GPRS and GSM. This two graphics shows protocol behavior when the poll answer waiting, time-out, is set less than the minimum required for the transmission delay of the communication channel (GSM or GPRS).

will be taken at least two seconds to poll one RTU. So the performance of the protocol will be reduced.

Table I represents the percent of errors because of timeout expired for a total of 500 polls over one RTU. For GPRS a timeout of 750 ms would be acceptable while on GSM it is needed timeouts over 1000 ms.

TABLE I

Time-out	500 ms	750 ms	1000ms
GPRS	100%	6,2%	3,2%
GSM	100%	80%	0%

The CC could poll simultaneously all RTUs using GPRS, but one by one sequentially using GSM.

In GSM to poll one RTU takes around 1 second. As it is not possible to poll more than one station at once, polling n stations will take n seconds, so the maximum number of RTU depends on the real-time application where the RTUs where used.

V. Conclusions

In this work the IEC 67080-5 application layer protocol over TCP/IP has been developed and tested for an open and flexible RTU. The RTU hardware is based on FPGA that has been programmed with the open core LEON with Linux operating system running over it.

The first IEC application profile for the RTU includes the poll function, that is, all data acquired from the RTU have to be specifically requested.

All the software design has been made in a PC platform using standard development tools. The source code generated for the protocol has been compiled with the standard Linux gcc compiler in LEON.

Two set of tests has been made. In the simulating test one CC and n RTUs has been all executed in one computer or two computers to determine the suitability for real time application. As shown, with a high number of RTU, 100 the maximum tested, there is a poll answer time of 0,9 ms quite enough for real-time, but the transmission delay is almost 0, so the previous claim also depends on the channel characteristics.

In the field testing, the RTU prototype with GPRS and GSM transmission channels has been tested. In this case the focus was on setting the appropriate poll answer waiting time (time-out). As has been shown, in the case of GPRS a time-out of 750 ms is enough to receive the poll answer, but in GSM the time required is 1000 ms. Although the time-outs are closed, in GPRS it should be possible to poll more than one RTU in this time, same scenario as the simulating test, however in GSM a RTU required 1000 ms, consequently each RTU have to be poll one by one and the maximum number of RTUs depends on the environment the RTU is applied.

Further researching work will be made to develop and test the IEC 60870-5 application layer but this time over the standard data-link layer, that is IEC 60870-5-2, developed and tested in [4] and [5].

Acknowledgments

This work has been undertaken in the framework of two research projects: OFU (EXC-2005-TIC-1023) - Open Flexible Unit funded by Junta de Andalucía and TOMARES (TEC2006-08430) -Multimedia Operatives Techniques applied to Supply Electric Networks funded by the Ministry of Education and Science of Spain.

REFERENCES

[1] Jaime Benjumea Mondéjar, Ana Verónica Medina Rodríguez, Isabel María Gómez González, Enrique Dorronzoro Zubiete, Gemma Sánchez Antón, Sergio Martín Guillén: Choosing the Right Protocol Stack for an Open and Flexible Remote Unit. ISIE'2008. International Symposium on Industrial Electronics. International Symposium on Industrial Electronics. Cambridge, Reino Unido. IEEE. 2008. Pag. 1668-1673. ISBN: 1-4244-1666-0.

[2] A.Muñoz, E.Ostúa, M.J.Bellido, A.Millán, J.Juan, D.Guerrero: Building a SoC for industrial applications based on LEON microprocessor and a GNU/Linux distribution. ISIE'2008. International Symposium on Industrial Electronics. International Symposium on Industrial Electronics. Cambridge, Reino Unido. IEEE. 2008. Pag. 1727-1732. ISBN: 1-4244-1666-0.

[3] International Electrotechnical Comission, "International Standard IEC-60870-5" (6 parts).

[4] Enrique Dorronzoro Zubiete, Isabel María Gómez González, Ana Verónica Medina Rodríguez, Jaime Benjumea Mondéjar, Gemma Sánchez Antón, Sergio Martín Guillén: Abstract Implementing Iec 60870-5 Data LINK Layer for an Open and Flexible Remote Unit. 34th Conference of the IEEE Industrial Electronics Society (IECON 2008). Num. 34. Florida, Orlando (USA). IEEE. 2008. Pag. 268-268

[5] Enrique Dorronzoro Zubiete, Isabel María Gómez González, Ana Verónica Medina Rodríguez, Gemma Sánchez Antón, Sergio Martín Guillén. Implementing Iec 60870-5 Data LINK Layer for an Open and Flexible Remote Unit. 34th Conference of the IEEE Industrial Electronics Society (IECON 2008. Num. 34. Florida, Orlando (USA). IEEE. 2008. Pag. 2471-2476

[6] Jaime Benjumea, Francisco Pérez, Joaquín Luque: "Encouraging the use of Open Source Software in high-sensitive environments", CIGRE, Study Committee D.2, Colloquium. Rio de Janeiro (Brasil), Sep, 2003.

[7] "GRLIB/LEON3 manual", http://www.gaisler.com

[8] A. Muñoz, E. Ostúa, P. Ruiz, M. J. Bellido, J. Viejo, A. Millán, J. Juan, D. Guerrero, "Un ejemplo de implemetación de una distribución Linux en un SoC basado en hardware Linux", Actas de las IV jornadas de computación reconfigurable y aplicaciones (JCRA'07), pp. 85-92, Sep-2007.

[9] "Wavecom Fastrack 1306M User Manual", http://www.wavecom.com/media/files/support/Hard_platforms/Modems/Fastrack_M1306B/User_manual/Fastrack_M1306B_User_Guide_rev003.pdf