Distributed Processing System for Gas Odorization

[1]Mário Reis, [2]Filipe Perdigoto, [2]António Pires
[2]Digiwest - Wireless and Embedded Solutions Lda
[1]REN-Gasodutos
Leira, Portugal
mario.reis@rengasodutos.pt
{filipe.perdigoto,apires}@digiwest.pt

Fernando Henriques, Sérgio Faria, Senior Member IEEE
Telmo Fernandes, Member IEEE
ESTG, Instituto Politécnico de Leiria
Instituto de Telecomunicações
Leiria, Portugal
fjorgehenriques@gmail.com
{sergio.faria,telmo.fernandes}@ipleiria.pt

Abstract— **This paper proposes an advanced industrial controller prepared for hazardous industrial enviroments, namely natural gas odorisation. Its design and conception is focused on robustness and reliability, with high flexibility to allow local and remote monitoring and control. Instead of a centralized approach, where the computational load relies on a unique processor, we propose a distributed architecture, where the various control tasks are distributed among the existing microprocessors, like in a DCS system, but yet exhibiting a fast response time between inputs and outputs, like a PLC.**

The proposed architecture presents a high degree of flexibility and scalability, in order to allow an easy addition of new modules. By resorting to a script language interpreter installed in each I/O board, they may be reconfigured to a different functionally, thus increasing the system reliability and safety. An efficient and synchronized communication between boards ensures that all memory boards share the system status information, thus allowing each board to process individually.

With the same performance and flexibility of a high-end PLC controller with full extras, like tactile HMI and TCP/IP remote control, we present an affordable gas odorizing system, which can be easily adapted for other industrial applications.

Keywords—Distributed system, Industrial controller.

I. INTRODUCTION

Gas produced from coal, besides other components, contained a wide range of sulfur compounds, which made it easily detectable in case of leaks and lent it the typical "gassy odor". This allows humans to recognize the smell and to be alert for the danger. But, natural gas is an odorless and colorless flammable gas, which originated serious accidents with many casualties in the early 20th century. As a consequence of these accidents the use of odorants was enacted, such that the gas is readily detectable by a person with a normal sense of smell. It means that the presence of natural gas in air must be detectable by smell [1].

Gas odorization refers to the operation involving addition of an odorant to gas, ensuring a characteristic odor of natural gas in order to give people a distinctive and unpleasant odor, so that the presence of gas in air in concentrations below the lower explosive limit (LEL) is readily detectable. Through the odorant addition, one cannot change any physical or chemical property of natural gas, except the smell. Generally for those who use natural gas, in the process of delivering for both public and industrial use, odorization provides safety. The main task of natural gas odorization is to ensure such operating condition when gas in every part of the distribution grid fulfills the requirement of a "warning odor level". In order to overcome such need, various equipment are used in the gas distribution chain to perform the odorization process, from odorization to monitoring. An odorizing system is comprised of electronic and mechanical parts. While the mechanical part actuate in the gas, the electronic equipment controls the mechanical actuators.

Being an expensive product, the odorant product added to gas should be kept at a minimum. At the same time, being also a safety issue, one cannot take the risk to sub-odorize the gas. This way, as the controlling element, the odorization controller plays a key role in gas odorization, as it calculates the required odorant product. This odorant level is based on the gas volume/flow, in order to maintain an ideally constant concentration of odorant. In case of some system component problem (e.g. lack of odorant product, pump malfunction, etc.), it is also the controller that must generate alarms, locally or remotely, as soon as possible. This will allow restoring the odorization system good functioning in a shortest period of time.

In this paper we focus only on the design and development of an odorizing controller, including hardware and software, based on a distributed processing architecture. A distributed system relies on a software system in which components located on computational devices communicate and coordinate their processes by passing messages, thus comprising a global system [2]. In these systems, the components interact with each other to attain a common goal. The principal characteristics of distributed systems are: concurrency of components, lack of a global clock, and independent failure of components. In distributed computing, a problem is divided into many tasks, each of which is solved by one or more computers [3], which communicate with each other by message passing [4].

Despite the technology advances in the last decades, natural gas odorization systems have not followed such level. Nowadays, due to the high standards of quality of service required by most companies, it is necessary to manage and supervise all systems in the production/distribution chain in a real (or near real) time basis. This only can be achieved if the equipment's in charge of operation allow such interaction.

Some of the identified weaknesses of the existing systems are related with to local interaction with the odorization equipment and others related with the remote monitoring of the system. Regarding the local human machine interface, usually it is very poor, allowing only minimal modification of values, namely the odorant mean value to be added to the gas, as can be seen in the portfolio of the major European players (Fiorentini and Lewa) [5, 6].

In industrial automation, systems are implemented using PLCs (Programmable Logic Controllers) [7] or, DCS (Distributed Control System), wherein control elements are distributed throughout the system by means of communications networks for command and monitoring connect a hierarchy of controllers [8]. Each processor receives information from input modules and transmit information to output modules. Additionally, input modules receive signals from input instruments in the process and the output modules transmit instructions to the output instruments. Both, input signals can be either analog or discrete signals. Electrical buses connect the processor and modules through the distributed controllers with the central controller and finally to the Human–machine interface (HMI). DCSs are commonly designed with redundant processors to enhance the system reliability. But, adequate deployment of hardware, software, and applications requires considerable knowledge and skill.

DCS systems, usually installed in large industrial plants, use a network to interconnect various processing units, instead of centralizing the process it in a unique controller. In comparison to a centralized system, as a PLC, a distributed system present some advantages like: capability to store data and create data bases; ease in ensuring redundancy on process execution; alarm control and management to allow failure detection and prevention; ease in implementing new features, among others.

Many systems present very strict response time requirements between inputs and outputs, being difficult and costly to implement. While PLC based systems are capable of presenting short time cycles, DCS systems usually have a high response time between input stimulus and outputs. The proposed controller includes the benefits of both systems, taking advantage of the modularity and flexibility of a distributed system, to support the integration of a data logger, HMI, communication system and redundancy in I/O (input/output) boards, without compromising the cycle time corresponding to monitoring and I/O actuating. It is targeted to take control of the many existing mechanical systems, but also allow evolving to more complex and secure systems.

The remainder of this paper is organized as follows: in Section II the requirements are presented, and the hardware and software design are presented in Sections III and IV, respectively. In Section V the prototype is shown and in Section VI some conclusions are drawn.

II. System Architecture

The controller architecture must provide a efficient processing and communication between the various modules, as represented in Figure 1. First an introduction to the designed hardware is presented, and then a description of the software architecture is given.

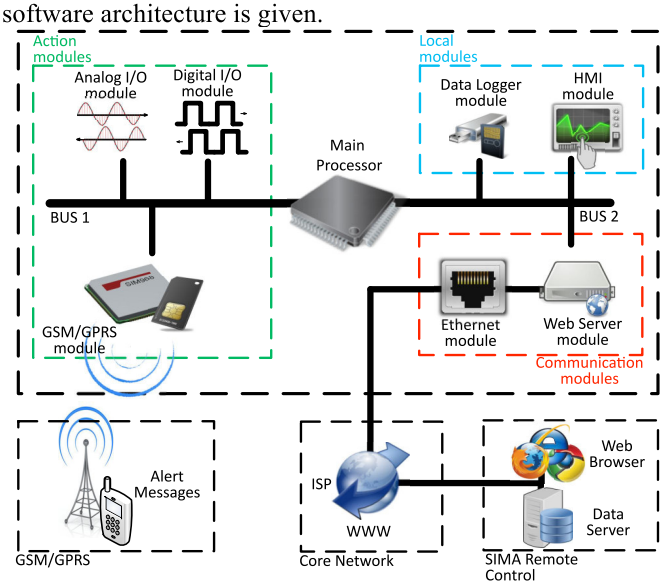


Figure 1 – Controller architecture.

A. System Hardware Architecture.

The success of any embedded system hinges on the performance of the underlying algorithms. However, the overall system performance highly depends on the hardware design. Consequently, the hardware boards were carefully designed to facilitate the application execution, not only by efficiently designing the interconnection between boards, but also optimizing the memory usage.

In order to alleviate the main processing board with signal conditioning and interfacing, all modules, except the power board, include its own processor with memory resources, which performs specific processing within a board and interfaces with other modules through a bus. Despite all used microcontrollers have, at least, two SPI (Serial Peripheral Interface) buses, only the main processor (in the main board) connects to both, SPI Bus 1 and SPI Bus 2.

The proposed architecture utilizes two SPI buses, as shown in Figure 2. SPI Bus 1 is exclusively dedicated to the most time-critical communications between I/O modules and the main CPU. The modules requiring a higher data bandwidth, like Ethernet, USB storage and HMI, are connected via SPI Bus 2. Nevertheless, both buses offer a transference rate of 10Mbps. The bus separation into two parts ensures the control system is not affected by possible data transference congestion due to high data demanding from HMI and storage modules. This way, time response of the control system may be kept as low as possible.

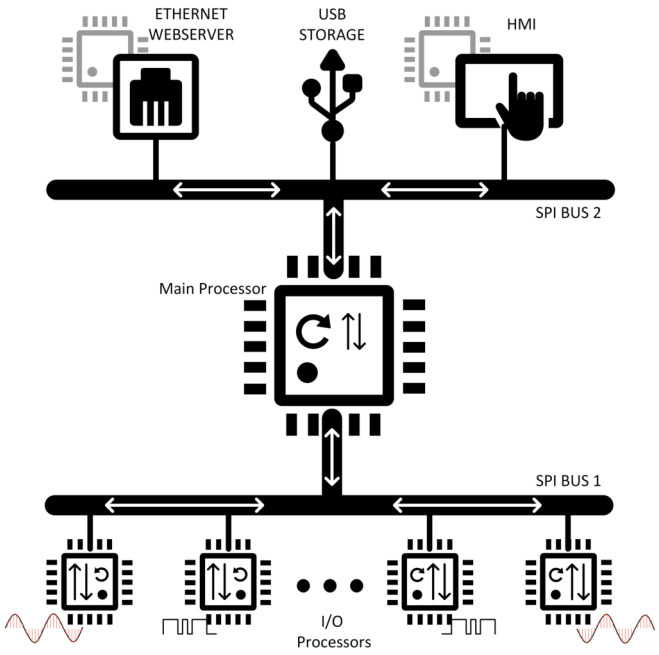


Figure 2 – Hardware architecture.

The **Main processor** is based on a microcontroller that forms the heart of the control system. In addition to implement the odorization control algorithm, this board is responsible for the management, prioritization and synchronization between all modules. During the initialization it scans the system to detect all the connected modules, establishing a running configuration accordingly. The main board, that performs the closed loop digital control, is based on a 32-bit microcontroller PIC32MX470 from Microchip [9], operating with a clock frequency up to 120MHz (1,86 DMIPS/MHz), 512 kB of Flash memory and 128 kB of High Speed SRAM. This microcontroller is also the master of both Serial Peripheral Interface (SPI) buses, SPI BUS 1 and SPI BUS 2, which allow sharing information between all modules.

When the system fails, this module activates the alarms, according to the security plans, registers the occurrence in Datalogger module, and generates the alarm through either dedicated digital discrete outputs, the Ethernet or the GSM, in order to minimize the technical intervention delay. Besides controlling the data flow between all modules, it implements an algorithm of proportional control, a type of feedback control system, to perform the odorization of gas.

A high precision **RTC** (real time clock) is used, with battery backup, which allows having an accurate real time clock all over the system, even if the system remains unpowered for several months.

The **DataLogger** module contains one SD card flash memory for high capacity storage. The SD card, externally available, will allow storing all operation data (operation values and alarms/events) for a period of several years. This flash memory implements a file FAT32 system system to handle an exhaustive registering of all relevant data for the system lifetime. This system will also be able to act as a USB host, allowing user to easily download all stored data in SD card to an external USB pen.

A **Flash Memory** circuit is used in order to store system configurations. This memory will also be used to store data that is accessed frequently, allowing this way high-speed access to key data.

The **Ethernet TCP/IP** implements a webserver using an additional dedicated microcontroller that implements a TCP/IP stack. This communication interface includes an Ethernet transceiver to support a wired connection, which may be linked to a central server or Internet.

This connection allows not only the access to the internal webserver for remote visualization and control of the local controller (remote virtualization of controller), but also to connect to a remote server that manages a network of GOC's (diagnostics functions, database, network integrated view, etc.).

The **I/O modules** comprise different types of digital and analog input/output circuits, which are based on a PIC32MX130, with a clock frequency up to 50MHz (1,66 DMIPS/MHz), 256 kB of Flash memory and 64 kB of SRAM.

- **Analogue I/O module** signals are based in the well-known industrial standard 0/4..20mA current loop. Analog external equipment, like transmitters, can be directly powered by analog I/O boards (24V/28mA), with no need of an external power supply. This way, both passive and active analog signals can be connected to system analog I/O's. Analog channels have a resolution of 16bit and were designed to have a maximum error of 1μA, with no need of adjustment. All analog channels are galvanically isolated from each other and from the main system, with a minimum isolation voltage of 2kV.
- **Digital I/O modules** include hardware to interface with external digital signals. The output signals can be solid-state or relay-based outputs.

  The relay-based outputs have the capacity to directly drive loads with currents up to 5A and voltages up to 220Vdc/250Vac, with a switching capacity up to 60W/62.5VA. User can also independently select the state of each channel relay contact in unpowered condition, by means of onboard logic selecting switches. Each channel has a replaceable 5x20mm protection fuse.

  The solid-state outputs can drive loads up to 50mA(dc)/35mA(ac) and withstand voltages up to 250Vdc/200Vac. Each channel is protected by an ultra-fast (<1μs) 75mA (nominal) auto-retriggerable electronic protection/fuse.

  The digital inputs were designed to work with PLC typical voltages levels, that is, 24V nominal, where, by design, input voltages below 5V are guaranteed to be 0 and input voltages above 10V are guaranteed to be 1. The digital input channels can tolerate input voltages up to 300Vdc/230Vac without causing any damage, and the input impedance is 60kΩ minimum.

  Additionally, due to the application environment, that is potentially explosive, the interfaced signals must be compliant with the ATEX directive. Such a digital I/O

module was also designed, but due to the certification process, at the initial stage, only external commercially available ATEX Intrinsically Safety barriers (IS barriers) were used [10].

- The **GSM/GPRS/UMTS** is an optional module that may be used like an I/O module. It circuit includes a microcontroller that implements the interface with the Main Processor via SPI with the GSM wireless network, as shown in Figure 3. It can be used to send alarms, whenever the connection through the Ethernet with the server is not available. This redundancy increases the level of security, allowing alert messages (e.g. in case of pre-programmed levels of product being achieved, pumps failure, or other user-defined events or problems occur in the odorizing system).

The **Backplane** module incorporates the power and signal lines to interface the required modules. The signal lines are divided into BUS 1 for the I/O modules, and BUS 2 for the HMI. The Main Processor is the master for all bus signals communications between modules.

The **Power Supply** module (PSU) is powered by a 24Vdc±15% source and is protected against under voltage and overvoltage. In case of an input under voltage (<20V) or overvoltage (>32V), the PSU will shutdown its outputs, until input voltage returns to acceptable values (20V...32V), preventing equipment from being damaged (input fuse is not affected). If an input voltage in excess of 80V is applied, the PSU implements a crowbar protection circuit that blows the input fuse, preventing it from being damaged.

The **HMI** (Human-Machine Interface) is composed of two modules: a local one with a 7" capacitive LCD with touch capabilities, and a remote connected through Ethernet. Both rely on a 32-bit microcontroller PICMX430, with a clock frequency up to 100MHz (1,31 DMIPS/MHz), 512 kB of Flash memory and 128 kB of SRAM. This module was designed to offer a pleasant user-friendly interaction. It is comprised of a 7 inches capacitive full color display. A second input device is also provided as a redundancy to the capacitive tactile display. This can be either a small 6-key capacitive keyboard or a jog shuttle wheel. This module has a main microcontroller that controls the graphical interface and manages all communication with the remaining system. If the 6-key keyboard is used, a second microcontroller is responsible by this keyboard management.

B. System Software Architecture

As mentioned before, the system is highly flexible, and the key for that flexibility is modularity, is not only in hardware, as previously explained, but also in software. The software has been implemented using a modular (layer) approach, where functions have been grouped into layers, allowing reusability among different implementations with minor, or even no changes. In general, it is comprised of two layers, a lower layer implementing the particular kernel adequate for each board type, and an upper level with memory updated with data form other boards, as can be seen in Figure 3.

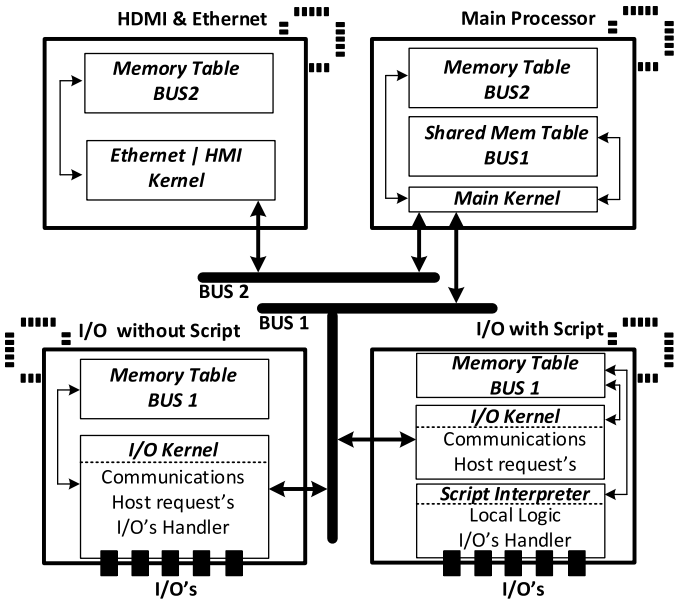


Figure 3 – Software architecture.

The SPI protocol has a master/slave topology, where one device (e.g. main board) has unidirectional control over one or more of the remaining devices (e.g. I/O boards). As a slave device cannot start data transference, each slave has an additional signal connected to an interrupt line of the main processor. Through this interruption, each slave module may request to start a communication with the main board, therefore increasing the efficiency by reducing redundant data acquisition of the main kernel.

The kernel is the program code that manages the system critical tasks, like I/O updates, bus communication management and data memory updates, and translates them into data processing instructions for the central processing unit and other electronic components on the board.

One of the most important aspects of the implemented system relies on the local memory tables for sharing information between boards connected by the bus. Each table maps all data and control variables required by the control system. These tables synchronized (equal) in all modules, and therefore if the user changes any parameter in the control system through de HMI, this modification will also take place seamlessly in each module due to the action of the distributed kernels.

I/O boards are all connected by SPI 1 bus, in order to avoid possible data congestion caused by an overlong data logger or HMI module request. These boards may be used in two topologies, with (Script) or without script (WScript) language interpreter, as exemplified in Figure 2. For the WScript topology, the I/O kernel is responsible for updating and synchronizing the memory tables, updating input and output data, and managing communications with the main processor. In this topology, the I/O board does not perform any logic processing related to its inputs and outputs, but sends information to the main processor and receives processed data

to actuate the outputs. The control processing is centralized in the processor of the main board.

In the Script topology, the I/O kernel does not control the system inputs and outputs, but it is responsible for the communications with the main processor, for synchronizing the memory table, and for updating the memory table fields not related to the system inputs and outputs. In this topology, the local processor executes the logic processing related to I/Os, not depending on the main processor. The processing algorithm will be supplied through a script that will be interpreted by the script interpreter installed in the program memory of the local processor. The script interpreter will substitute the I/O kernel in the updating of the memory fields related to the I/O board inputs and outputs. Besides the script interpreter installed in the I/O boards processor, the HMI module supports a script editor to allow users to modify and adapt each I/O board to a new algorithm, depending on the application. There are few script interpreters compatible with 32-bit Microchip PIC32MX microcontrollers, for instance the interpreter MMBasic [11] for BASIC language, that is written in ANSI C and requires 94 kB of flash memory and approximately 16 kB of RAM. Once using the script interpreter, the I/O board processor is able to execute processes independently from the main processor. However, this increased functionality slightly affects the performance, as each BASIC command is executed in approximately 35us, thus being slower than a native instruction.

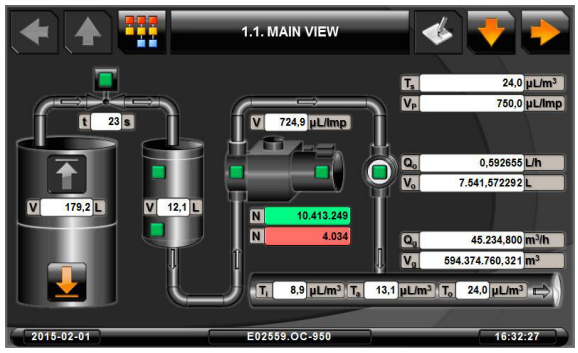
The script implementation, and consequent independency from the main processor to execute the local process, not only alleviates the main processor from a computational burden, but also adds new capabilities in terms of safety and redundancy. Redundancy of processes is commonly used when a system failure have unbearable results for the system. However, redundancy is costly and should be avoided when not strictly required. Through an I/O board programed with scripts it becomes simpler to implement redundancy, for instance letting an I/O board to be reprogramed with a script to replace another module under failure, thus overcoming the system malfunction prior to hardware intervention.

C. Application level software

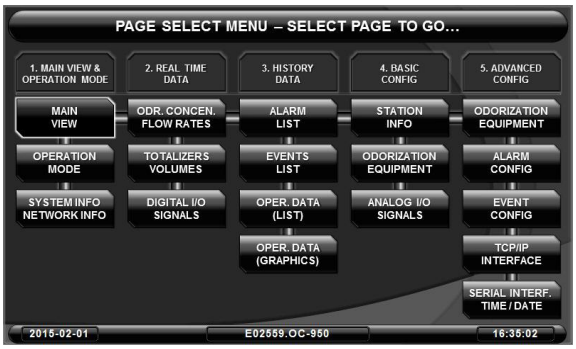
In the application level, the main algorithm is implemented accessing data from the memory tables and writing the results into these tables. While in most modules there is a simple algorithm to manage its specific activity, the Processing board hosts the main algorithm to perform the odorizing process. This is the most important task; therefore it is described with more details. Based on the study of other controllers and on odorization technician's inputs, a set of new functionalities has been implemented. In this type of system, the main objective is to keep constant, and controlled, the odorizing concentration in the gas. In order to achieve this goal, an odorizing injection pump must be accurately supervised, usually by means of a controller board. In this sense, the proposed system includes a processing board, accommodating a robust control algorithm where the flow/volume of injected odorant product is proportional to the supplied gas flow/volume. Thus, achieving an odorant concentration almost constant and equal to the user pre-defined value, even in the presence of large gas flow fluctuations.

The developed algorithm also supports gas flows above nominal values, namely peaks of gas consumption that may sporadically take place. When such events occur and the injection pump is unable to follow it, the unattended commands are stored in memory and executed as soon as the gas flow conditions allow it. The algorithm was developed to allow a great stability of the odorizing control, even in the presence low flow of gas/odorizing levels. Additionally, for security purposes, a functionality to detect a failure on the gas measurement system was implemented. In such occurrence, by default the controller board determines a set of values to substitute the missing measured gas flow values. Instead of simply substituting the odorizing parameter by a user-defined value, which is also an option, this algorithm is innovative. Due to the high storage capacity of the system, there is a long historical of the odorizing process, that is 24 values per day, one per hour, seven days a week. This historical data will allow the algorithm to select the best odorizing profile previously occurred; based on the gas flow occurred until the metering failure.

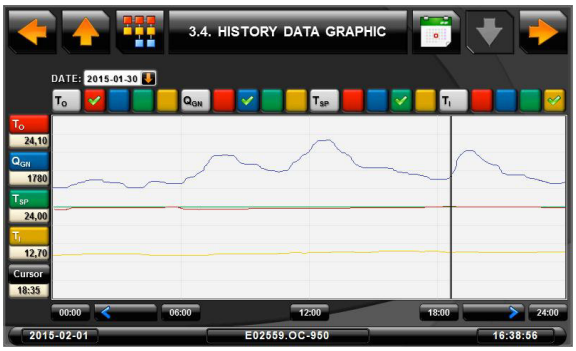
Another important feature that has been performed for this system is the HMI design and programming. Despite it has a professional look; the HMI has not been conceived using commercial tools for graphic design. The authors have developed the HMI firmware using objects approach. It is the case of all graphic objects, than can be easily reused in different implementations. The results can be seen in Figure 5, in this case, the version to be used with the field equipment of a major European manufacturer. Figure 4.a presents the main view/main screen, where a resume of all relevant operation values of the system can be seen at one single glance, unlike traditional systems, that scatters this information among several different text screens. Figure 4.b presents the Page Select Menu, where user can see all pages available, allowing a direct jump to any one of them. In figure 4.c one can see time based graphics of 4 different operation data sets, using the same time base. This way, a technician can see the time correlated variables, allowing a more easily diagnostics of an eventual system malfunction.



a) Main View.



b) Page selection Menu.



c) Data graphics.

Figure 4 – HMI display and tactile interface.

For processes requiring high refreshing rates, like the I/O modules, the use of a script interpreter is recommended as it allows moving the logic processing of each module from the main processor to the local processor located in each board. In the limit, this results in shortening the response time by the time spent in communications.

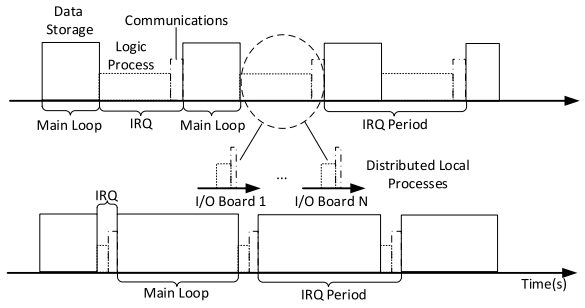


Figure 1 - System Kernel on IRQ + I/O Scripts processing

As illustrated in Figure 5, theoretically, it would be possible to remove all logic processing from the main board to the corresponding I/O module, just maintaining in the main processor the interrupt processes related to the modules connected to the SPI 2 Bus. Using this approach, communications in SPI 1 Bus are strongly reduced, as the main processor only will require new data from I/O boards to update the graphical interfaces (HMI).

This system has been implemented and tested for an industrial application, namely an odorizing gas system. However it can easily be used for other applications, even with very strict time response requirements.

III. Discussion and Conclusions

In this paper we have proposed a flexible electronic controller for an odorizing system, based on a distributed processing architecture. This architecture presents a performance and flexibility of a high-end PLC or DCS controller with full extras, like tactile HMI and TCP/IP remote control at an affordable cost. With the adoption of the different methods and techniques it was possible to achieve a performance of a DCS controller, while achieving PLC time response performance, for example from 500 ms to 10 ms.

By using separated buses it was possible to avoid data congestion caused by modules that deal with higher volume of data, namely for storage purposes and visualization. The second bus, free from this possible source of congestion, allows ensuring that data is transferred at the pre-defined period, thus increasing the system reliability. Messages transmitted between modules are stored into memory tables, which are duplicated in all local boards. By using interrupts to synchronize the critical processing tasks, the accurate updating periodicity of system inputs and outputs is ensured.

Finally, the addition of script programming to the I/O boards permits to alleviate the computational load of the main processor, by executing these processes in the local processor. This not only speeds up the response time from input data to output, but also increases the overall reliability of the system. In case of module failure it is possible to easily allocate a task to another module, thus increasing redundancy and safety in the system, as required in most industrial systems.

CKNOWLEDGMENT

The authors acknowledge the support of Research supported by "Projeto ID&T nº2014/33684 Sistema Industrial Modular Avançado" and Digiwest – Embedded and Wireless Solutions, Lda.

References

[1] Daniel Tenkrat, Tomas Hlincik and Ondrej Prokes, Natural Gas Odorisation (Book Style), Primoz Potocnik (Ed.), 2010, ISBN: 978-953-307-112-1.

[2] Couloris, George, Jean Dollimore, Tim Kindberg, Grodon Blair (2011). *Distributed Systems: concepts and Design* (5th Edition). Boston: Addison-Weslay. ISBN 0-132-14301-1.

[3] Philip Brighten Godfrey, *Designing Distributed Systems for Heterogenity*, PhD Thesis, Univ. of California, Berkeley, USA, 2009.

[4] Andrews, Gregory R. (2000), Foundations of Multithreaded, Parallel, and Distributed Programming. ISBN 0-201-35752-6.

[5] Odomatic Fiorentini. Available from: <http://www.fiorentini.com>. [14 February 2015].

[6] Lewa - LEWA Odorizing systems. Available from: <http://www.lewa.com/en/solutions/industries/gas-odorisation/>. [14 February 2015].

[7] M. A. Laughton, D. J. Warne (ed), Electrical Engineer's Reference book, 16th edition, Newnes, 2003, Chapter 16 Programable Controller.

[8] Galloway, B., Hancke, G. P.,*Introduction to Industrial Control networks*, IEEE Comm. Surveys and Tutorials, Vol. 15, Issue 2, 2012.

[9] Microchip PIC32MX5XX/6XX/7XX Family Data Sheet (04/22/2013). Available from http://ww1.microchip.com/downloads/en/DeviceDoc/61156H.pdf

[10] ATEX Directive 94/9/EC on equipment and protective systems intended for use in potentially explosive atmospheres (ATEX).

[11] MMBasic: software. Available on from http://mmbasic.com/.