HARDWARE RECONFIGURABLE CONTROL WITH OPENED ARCHITECTURE FOR MOBILE ROBOTS

Leonimer Flávio de Melo
Laboratory of Integrated Automation and Robotics
FEM-DPM-UNICAMP (www.fem.unicamp.br/~lar)
leonimer@fem.unicamp.br

João Maurício Rosário
Laboratory of Integrated Automation and Robotics
FEM-DPM-UNICAMP (www.fem.unicamp.br/~lar)
rosario@fem.unicamp.br

Abstract. *With the fast innovation of the hardware and software technologies in embedded systems area, with application in the robotics and automation, more and more it becomes necessary the development of applications based on methodologies that facilitate future modifications, updates and increments in the original projected system. In this way, this article presents a system of opened architecture, distributing the several control actions in growing levels of complexity and using resources of reconfigurable computing proposal oriented to embedded systems implementation. Software and the hardware are structuralized in independents blocks, with connection through common bus. Is presented the functional blocks where the use of microcontroller is illustrated for local control level. The supervisory control level is implemented in an IBM PC platform and is connected with the local control level, in the robot, through Ethernet WI-FI link. Also are seen the control blocks that utilize the programmable logical devices (PLD) that are hardware projected for sensors fusion control interface and actuators controllers. The sensors, actuators, RF transceiver unit, and others necessary peripheral components for the project implementation with their implementation blocks are listed below.*

Keywords *Embedded Systems, Mobile Robots, Hardware Reconfigurable Systems, Opened Architecture Systems.*

1. Introduction

Platforms for knowledge consolidation in several teaching and research areas, such as modeling, control, automation, power systems, sensors, transmission of data, embedded electronics and software engineering are a necessity in teaching and research institutions. The use of the mobile robots for this purpose appears to be quite an attractive solution. It allows the integration of several important areas of knowledge and a low cost solution, which has already been adopted with success by other research institutions. Finally, as each day goes by, it becomes a better solution for practical problems in our society (Milk, 2005, Reese, 2002, Borenstein, 1995).

The proposal and development of this open and generic system aims at supplying this need, having as an emphasis, the control structuring, the supervision and the transfer of information. The development of this system demands the knowledge in terms of project aspects and integration that would not be approached if a commercial mobile robot was acquired.

Within the proposal of platform mobile robotics, the use of an embedded processor, with control software especially developed for the necessary applications is considered. Together with this, a commercial platform is analyzed, which coupled to a communication net, allows the creation of a powerful link with the external world. The objective of this platform is to make use of the existing communication interfaces, as well as to provide an embedded user interface alternative in the mobile robot. Another aspect considered, is the flexibility of the hardware project, which allows the expansion of mobile robot facilities. New sensor combinations should be used. Different supervision and control models should equally be used to carry out the mobile robot tasks.

2. The Programmable Logic Devices

Today's Programmable Logic Devices are no longer similar to their ancestors from early 90s. Changes in the manufacturing technology and the feature sets offered by the leading CPLD and FPGA vendors have been more revolutionary than evolutionary, especially over the past few years. Deep submicron processes, reaching 90nm (and the cutting-edge 65nm announced), ensure enormous capacity of the high speed logic fabric available to users. Besides the growing logic density, contemporary FPGAs offer specialized Digital Signal Processing units like Multipliers and Multiply-Accumulate circuits (MACs). With the introduction of industry standard RISC processor hard macros like PowerPC^{TM} 405 in Xilinx's Virtex II Pro family and ARM^{TM} 922T in Altera's Excalibur devices, FPGAs enter the world of System-On-Chip (SOC) designs where software co-exists, coordinates and cooperates with the hardware on a single silicon die. Both Altera® and Xilinx® also offer soft-core processors (mapped into programmable resources of FPGAs) that can be custom-tailored to the needs, requirements and capacity limitations. Configurable, high speed, built-in serial transceivers complying with several protocols simplify process of data transmission between devices and subsystems.

The flexibility of programmable devices is yet another advantage that should not be underestimated. The hardware implemented in a PLD can be upgraded while it is already in the end-user product. Static reconfiguration is a great advantage that allows for functionality update and bug removal without physical modification of the circuit structure. Moreover, the process of structure modification can be performed dynamically, while the system is operating. A dynamic reconfiguration allows for rational hardware utilization for the appropriate task. Only required processing components are used. Dynamic reconfiguration seems to be the most challenging area in hardware-software co-design (Milik et al. 2005).

The features mentioned above alongside continuously dropping prices of programmable chips open the doors for FPGAs and CPLDs in applications that were reserved for ASICs in the past. To empower the designers with the ability to efficiently use the capabilities of modern devices, new techniques and tools for design and verification are necessary. Some of these will be presented in the following sections of this paper.

In general, each FPGA device must be programmed before use. Depending on the configuration storage this process can be executed once in the life time (OTP) or the device can be reprogrammed. Devices based on SRAM configuration memory must be configured each time after power-on. Commands to reconfigure circuit and load new configuration data can also be issued during operation. The ability to change configuration data has an important implication – it allows updating the functionality of the circuit without changing its physical structure. When the configuration circuitry supports partial modification of configuration memory contents, it is possible to create a run-time re-configurable circuit. Circuits, similarly to computers with exchangeable programs, can be modified according to the current requirements. This allows reducing the logic capacity of the circuit by providing only a subset of the functionality – such that is required to execute the current task (which can be a small subset of all executed tasks).

In the past concept of the design was divided into several steps that were unlinked together. Refining the design from one abstraction level to another requires an extremely high effort from the design team. Models developed for a given stage of abstraction can hardly be used in other abstraction levels. The design starts from the concept modeled with the use of high level languages (C, Fortran, Pascal,…) or tools (Matlab, Mathcad,…). This level of abstraction allows obtaining algorithmic verification in early stages of the design. Verification in the domain of algorithm only assures the designer that a proper algorithm was chosen. In fact at this stage it is difficult to estimate the performance of the system while the performance can be impacted by several factors that are design dependant. An extremely important step after algorithmic verification is the design partitioning. An improper design partitioning can lead to inefficient implementation that hardly meets design constraints. Usually, during partitioning there are several factors that are taken into account. In general algorithms implemented in hardware are better than software competitors. Software implementation and debug process is much easier and less time consuming than for hardware. A properly partitioned design should have flexible hardware structures that allow assuring required performance with the help of the software platform. When this is combined with a system level approach, it is obvious that a specific approach is required for design development.

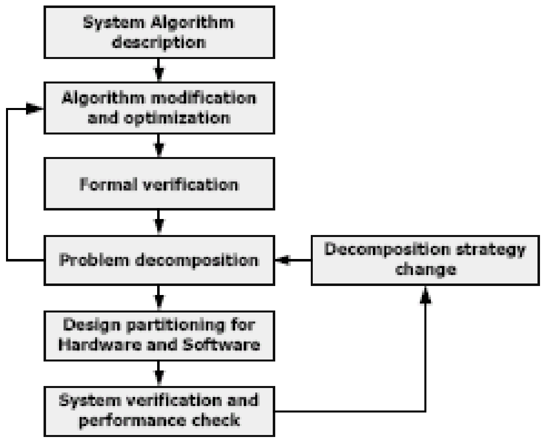

System Algorithm description
Algorithm modification and optimization
Formal verification
Problem decomposition
Decomposition strategy change
Design partitioning for Hardware and Software
System verification and performance check

Figure 1 - System level design flow

Contemporary designs are based on the system level paradigm (Teoh, 2002, Compton, 2002). System level design techniques are based on ready system components with a custom design part. The unmodifiable part of the system usually consists of a microprocessor system with a basic set of peripheral units used to solve a given problem (automotive, industrial, telecommunication, household applications etc.). Specific features of the designed product are located in the custom part of hardware and of course in the dedicated software.

The custom part of the hardware can be implemented as a mask programmed part of the chip or as an FPGA configured specifically for design needs. A very important factor in the system level design is performance. The performance problem is to a large extent dependant on placing a boundary between the hardware and the software part of design. This boundary is constrained by several factors like power consumption, FPGA part size, system performance, etc. In order to achieve best possible results, the partitioning process should be performed for different combinations of possible solutions (Fig. 2). The presented system solution requires a new design approach that allows a coherent design and verification of the product during each stage of the development.

As already mentioned, high level programming languages are used for algorithmic verification purposes. C language is extremely popular in the world of programmers, embedded system designers and hardware engineers. C language can be treated as a universal platform for program implementation and design verification. Universal also means unconstrained. In general, C language has not implemented any mechanisms to model system behavior contrary to HDL languages (concurrency, signal resolution, events).

On the other hand, a lot of high levels programming language structures are inherited by HDL languages. The gap between general programming languages and system design tool requirements can bridged by specific libraries that introduce system and hardware behavior implemented directly in C++ with specific classes operates as a testbench, then can be used during the entire lifetime of the design to verify different abstraction levels with the same transaction.

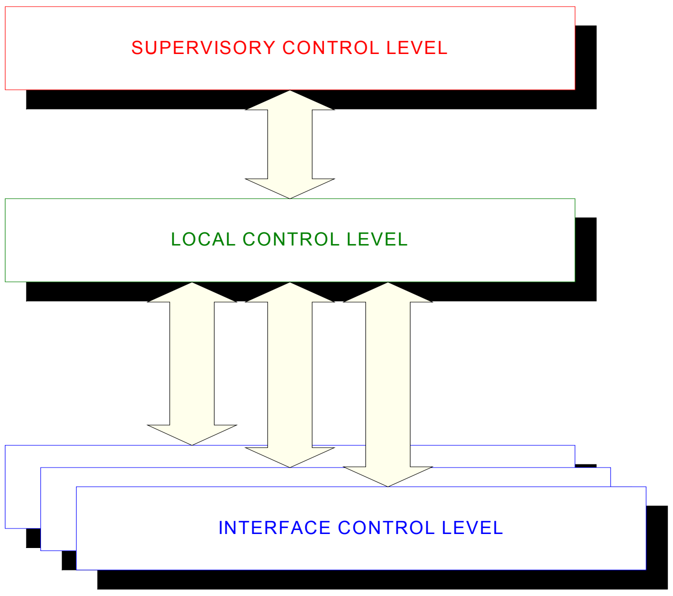


Figure 2 - Different control levels of the proposed system.

3. Proposed system

The proposed system can be visualized at a logical level in the blocks diagram in fig. 3. The system was divided into three control levels, organized in the form of different degrees of control strategies. The levels can be described as:

- **Supervisory control level:** This represents a high level of control. In this level it was possible to carry out the supervision of one or more mobile robots, through the execution of global control strategies.
- **Local control level:** In this level control was processed by the mobile robot embedded software implemented in a 8 bits microcontroller. The control strategies allowed decision making to be done at a local level, with occasional corrections from the supervisory control level. Without communication with the supervisory control level, the mobile robot just carried out actions based on obtained sensor data and on information previously stored in its memory.
- **Interface control level:** This was restricted to strategies of control associated with the interfaces of the sensor and actuators. The strategies in this level were implemented in hardware, through PLD (Programmable Logic Devices).

Architecture, from the point of view of the mobile robot, was organized into several independent blocks, connected through the local bus that is composed by data, address and control bus (fig. 3). A master block manager operates several slave blocks. Blocks associated with the interfaces of sensors and actuators, communication and auxiliary memories were subjected to direct control from the block manager. The advantage of using a common bus was the facility to expand the system. Inside the limitations of resources, it was possible to add new blocks, allowing an adapted configuration of the robot for each task.

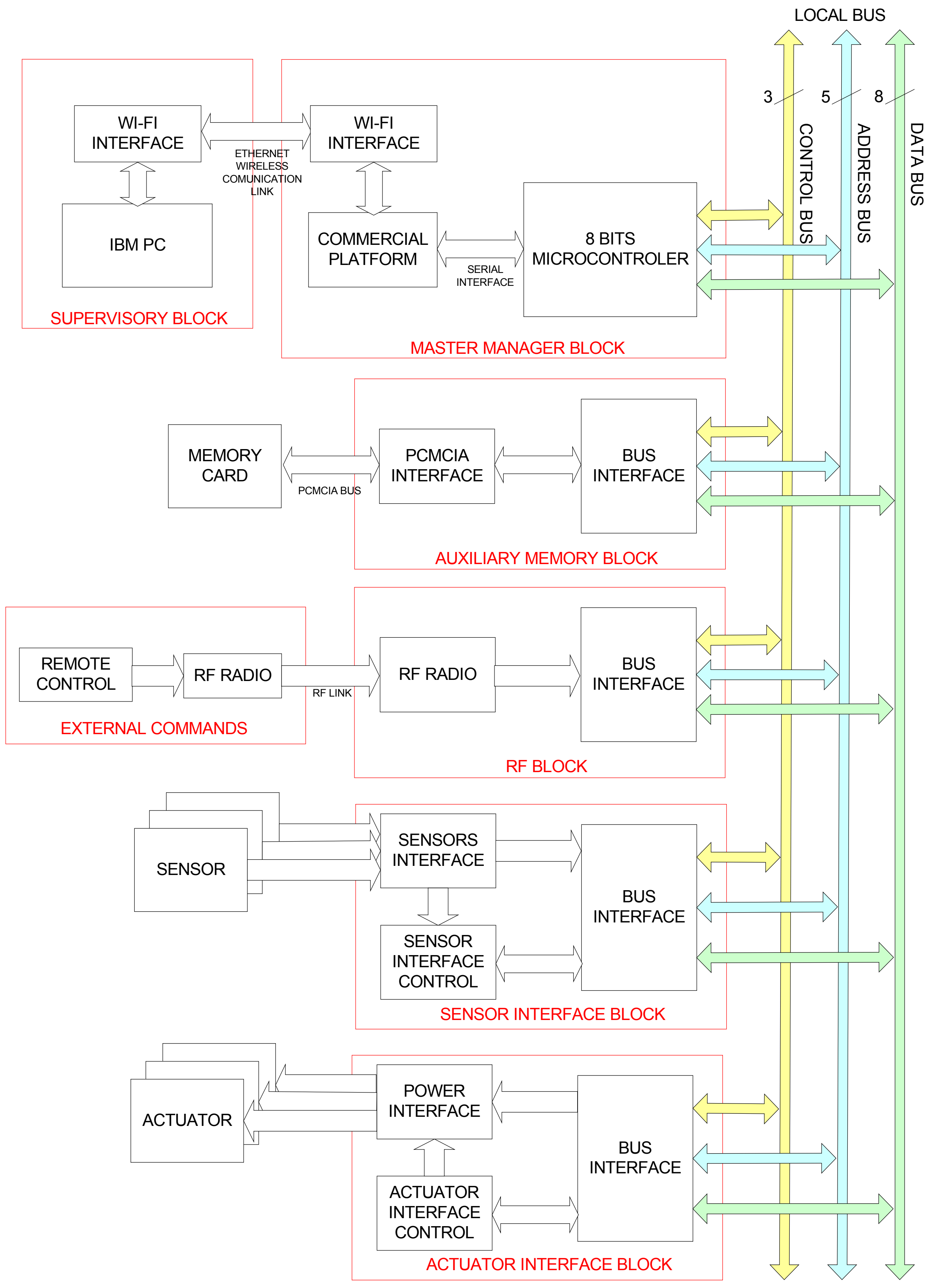


Figure 3 – Hardware architecture block diagram of the proposed system.

3.1. Description of blocks:

Supervisory control block: Is the high level of control. In this block is managed the supervision of one or more mobile robots, through the execution of global control strategies. Is implemented in an IBM PC platform and is connected with the local control level, in the mobile robot, through Ethernet wireless WI-FI link. This protocol uses IEEE 802.11a standard for wireless TCP/IP LAN communication. It guarantees up to 11Mbps in the 2,4GHz band and requires fewer access points for coverage of large areas. Offers high-speed access to data at up to 100 meters from base station. 14 channels available in the 2.4GHz band guarantee the expandability of the system with the implementation of control strategies of multiple robots.

Master manager block: Responsible for the treatment of all the information received from other blocks, for the generation of the trajectory profile for the local control blocks and for the communication with the external world. In communication with the master manager block, through a serial interface, a commercial platform was used, which implemented external communication using an Ethernet WI-FI wireless protocol. The robot was seen as a TCP/IP LAN point in a communication net, allowing remote supervision through supervisory level. This commercial platform operated as integral part of the mobile robot. Commercial platforms of low weight and with good battery autonomy were considered. There are several examples based on Windows CE operating system.

Sensor interface block: Is responsible for the sensor acquisition and for the treatment of this information in digital words, to be sent to the master manager block. The implementation of that interface through PLD allowed the integration of information from sensors (sensor fusion) locally, reducing manager block demand for processing. In same way, they allowed new programming of sensor hardware during robot operation, increasing sensor treatment flexibility.

Actuator interface block: This block carried out speed control or position control of the motors responsible for the traction of the mobile robot. The reference signals were supplied through bus communication in the form of digital words. Derived information from the sensor was also used in the controller implemented in PLD. Due to integration capacity of enormous hardware volume, PLD was appropriate to implement state machines, reducing the need for block manager processing. Besides the advantage of the integration of the hardware resources, PLD facilitated the implementation and debugging. The possibility of modifying PLD programming allowed, for example, changes in control strategies of the actuators, adapting them to the required tasks.

Auxiliary memory block: This stored the information of the sensor, and operated as a library for possible control strategies of sensors and actuators. Apart from this, it came with an option for operation registration, allowing a register of errors. The best option was an interface PCMCIA, because this interface is easily accessible on the market, and being a well adapted for applications in mobile robots, due to low consumption, little weight, small dimensions, high storage capacity and good immunity to mechanical vibrations.

RF communication Block: It allowed the establishment of a bi-directional radio link for data communication. It is used for the mobile robot to receive operation commands, allowing the learning of trajectories, for example. It operated in parallel with the commercial platform WI-FI link. The objective of these communication links was to allow the use of remote control. All the commands executed from the remote control are analyzed from master block manager to guarantee the best trajectory strategy. The remote control has a high trajectory priority from other blocks, like supervisory control block, and can take the control of the mobile robot to execute, for example, emergency necessary movements or stop.

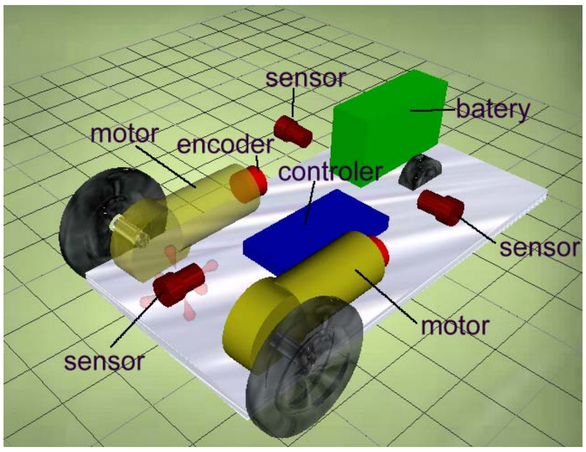
To implement this block was used a low power UHF data transceiver module BiM-433-40. It is a miniature UHF radio module capable of half duplex data transmission at speeds up to 40 kbit/s over distances of 30 meters "in-building" and 120 meters open ground. The module was chosen because it integrates a low power UHF FM transmitter and matching super- heterodyne receiver together with the data recovery and TX/RX change over circuits to provide a low cost solution to implementing a Bi-directional short-range radio data link.

3.2. Example of application

This architecture was sufficiently generic to be used with several models of mobile robots. As an initial validation, the use of a platform with a differential drive configuration was proposed (Mirab et al, 2002). A great number of sensors could equally been used, but two types were initially considered:

- Position sensor: Two encoders coupled to the traction axis of the robot. The encoders were responsible for the odometer registration as well as the determination of the mobile robot's displacement axis.
- Range finder sensor: Responsible for the distance measured to obstacles in front and on the side of the mobile robot.

Fig. 4 presents a model of the proposed platform implemented at UNICAMP, Laboratory of Automation and Robotics. The encoders were coupled directly to the axis of the motor, allowing for a larger resolution in odometer measurements. The range finder sensors were distributed to cover the probable areas of collision.



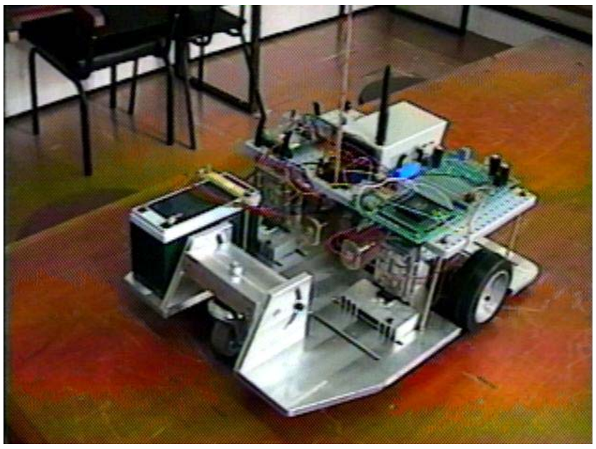


Figure 4 – Constructive proposal of the mobile robot's initial validation (Erig Lima et al, 2003).

An example of flexibility provided by PLD was illustrated in the speed and position control of the traction axis of the mobile robot, the aim of which is a DC motor control powered by a PWM drive. This programmable controller worked directly with digital signals generated by the encoder coupled to the motor and generated by the local control level, representing a trajectory (fig. 5).

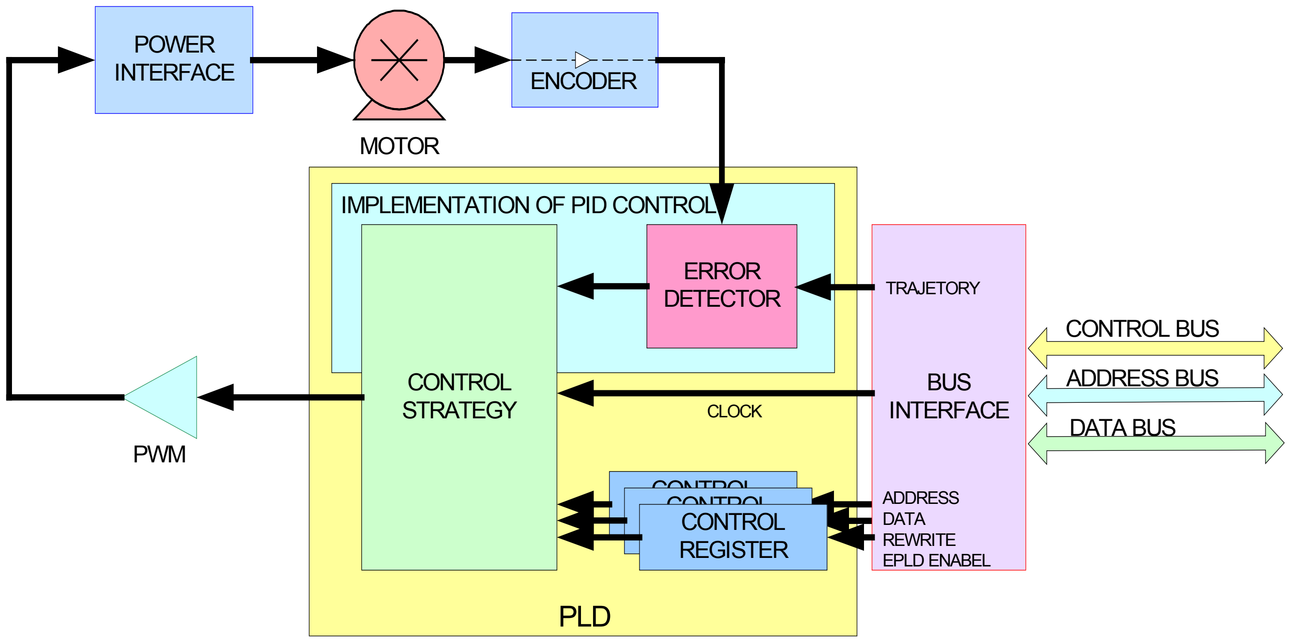


Figure 5 – Motor controller blocks - Diagram of the mobile robot.

Fig. 6 represents the proposed controller, with the various proposed control levels. PLD allowed this controller to be implemented in different ways, applying different control strategies. The controller can be, for example, a PID, an adaptive controller (Aström and Wittenmark, 1997) or a predictive controller (Soeterboek, 1992). In this case, information coming from the encoder direction or from the range finder sensor can be used as additional signal to the controller.

3.3. Supervisory control structure

Considering a final integration proposal and tests of the system in the proposed example the following stages were considered:

- Integration and tests of the manager block and the sensor and actuators interface block. Routines for verification of all the blocks were carried out. In this phase, the mobile robot is already capable of carrying out a great number of tasks.
- Integration and tests of the auxiliary memory block, allowing the registration of information from the robot operation for future debug process.
- Alternatives of control strategies were also stored. In this stage, the load of PLD during the operation of the robot was tested.
- Integration and tests of the RF communication block, allowing the remote transmission of data from the robot. In this stage the mobile robot's remote control was tested, which allowed the learning of trajectories, for example.
- Integration and tests of the commercial platform, allowing strategies of supervisory control of the mobile robot.

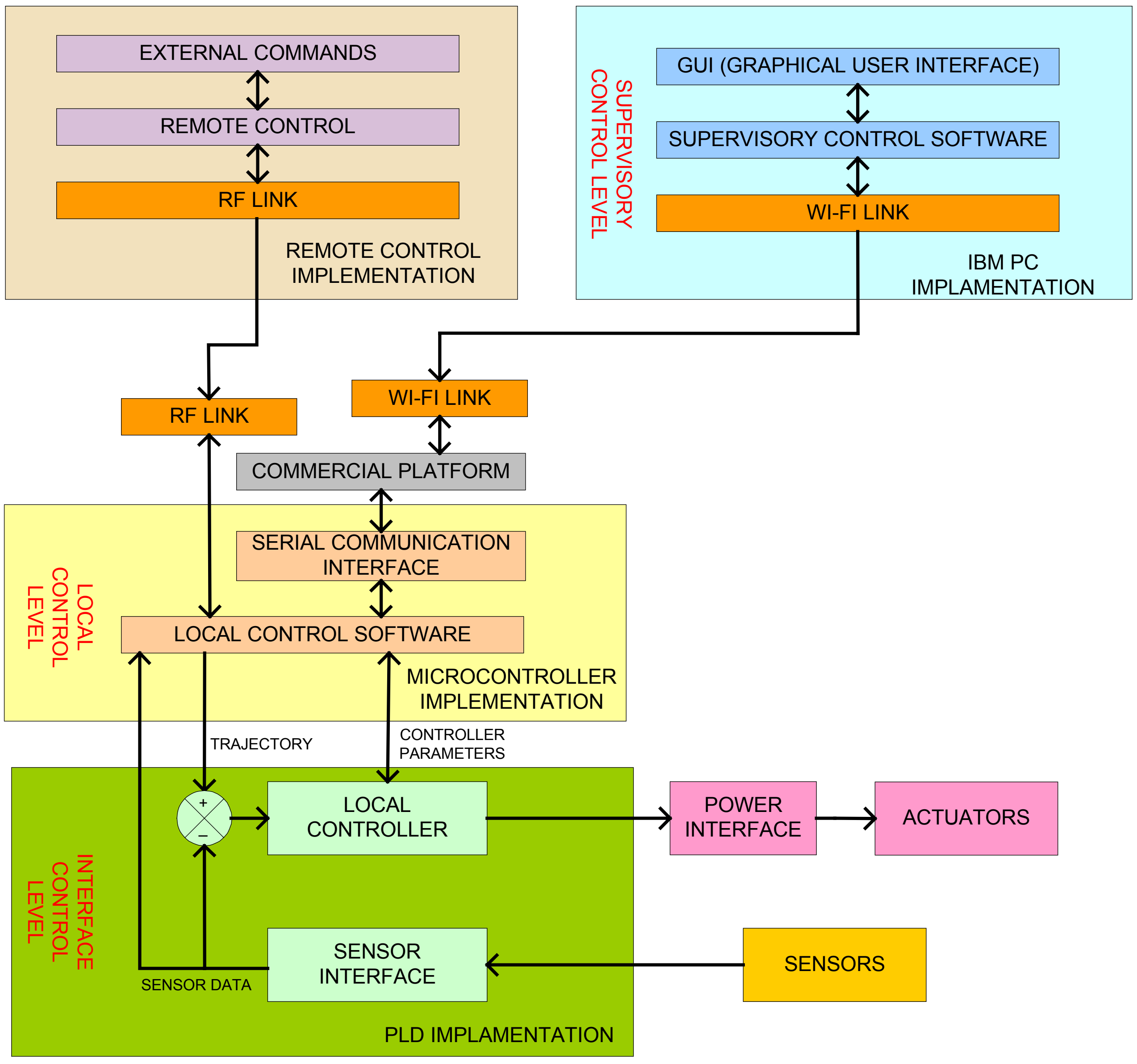


Figure 6 – Blocks Diagram representing the control levels structure of the mobile robot.

4. Conclusions

The main objective of this work was to propose a generic platform for a robotic mobile system, seeking to obtain a support tool for under-graduation and graduation activities. This came from encountering the growing need to propose to the graduated student's research that integrates the knowledge acquired in several disciplines that stimulates teamwork in order to reach a result. In graduate courses, this platform became an alternative support for several types of research, such as validation of control and supervision strategies, model generation and data transmission protocols, among others. Another objective was to gather knowledge in the mobile robotic area, aiming at presenting practical solutions for industrial problems, such as maintenance, supervision and transport of materials. Some promising aspects of this platform were:

- Flexibility - there was a great variety of possible configurations in the implementation of solutions for several problems associated with mobile robots.
- Great capacity of memory storage allowing implementation of sailing strategies for maps.
- Possibility of modification of control strategies during the operation of the mobile robot in special mechatronics applications (wheel chair, inspection robots), fig. 7.

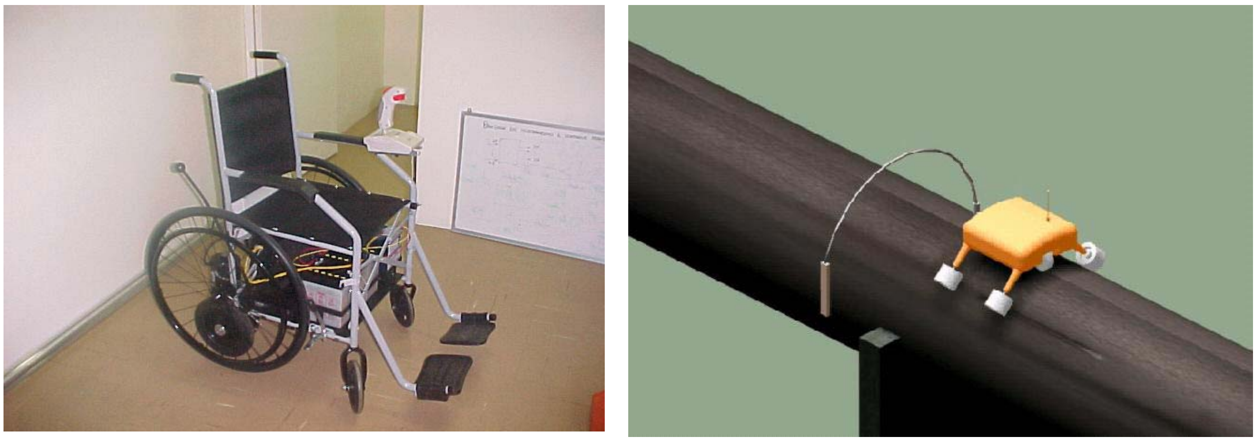


Figure 7 – Others applications - wheel chair, pipe inspection robots.

4. References

Aström, K. J., Wittenmark B., 1997, "Computer Controlled Systems", Third Edition, Prentice-Hall, New Jersey.

Borenstein, J., 1995, "Control and Kinematics Design of Multi-Degree-of-Freedom Mobile Robots", University of Michigan, Ann Arbor.

Compton, K., 2002, "Reconfigurable Computing: A Survey Of Systems and Software", ACM Computing Surveys, pp.171-210, vol 34, n. 2.

Erig Lima, C.R., Rosário, J.M., Ferasoli, H. Pegoraro, R., 2003, "Embedded Architecture Proposal Applied to Mobile Robots", Anais do VI Simpósio Brasileiro de Automação Inteligente – VI SBAI, pp. 602-607, UNESP, Bauru, SP.

Karger. A., 2003, "Architecture singular planar parallel manipulators. Mechanism and Machine Theory", 38, pp. 1149–1164.

Milik, A., Dykierek, M., 2005, "Tools and Technologies for Designing Control Systems Using Programmable Logic Devices", IFAC Conference, pp. 1-6

Melo, L.F., Rosario, J.M., Saramago, M.A.P., Erig Lima, C.R., 2005, "A Reconfigurable Control Architecture for Mobile Robots", Proceedings of MUSME 2005, the International Symposium on Multibody Systems and Mechatronics, Uberlandia, Minas Gerais, Brazil.

Miyazaki, T., 1998, "Reconfigurable Systems: a Survey". IEEE Proceedings of the Design Automation Conference 1998, ASP-DAC '98, pp. 447 - 452.

Mirab, H., Gawthrop, P.J., 2002, "Transputers for robotic control" - Parallel Processing in Control, the Transputer and Other Architectures, IEE Colloquium, pp. 330-336.

Reese, B., 2002, "Use of VHDL synthesis in an advanced digital design course" - Southeastcon '02, Proceedings., IEEE, pp. 509 - 512

Teoh, E. K., Yee, Y. E., 2002, "A DSP-based adaptive controller for robotic control application", Systems, Man, and Cybernetics, 2002. 'Decision Aiding for Complex Systems, Conference Proceedings, IEE Colloquium, pp. 961-965.

5. Responsibility notice

The authors are the only responsible for the printed material included in this paper.