Shop-floor controller based on RT-middleware technology

Ferenc Tajti[1,2], Géza Szayer[1], Bence Kovács[1], Péter Korondi[1]

[1]Department of Mechatronics Optics and Engineering Informatics, Budapest Univ. of Technology and Economics
[2]MTA-ELTE Comparative Ethology Research Group (MTA: 01 031)
Budapest, Hungary
ferenc.tajti88@gmail.com

Abstract— **Nowadays the flexible configuration of manufacturing cells becomes to an important requirement especially at small and medium sized companies. This method can make the production fast and effective at small series or frequent manufacturing changes. The shop-floor control method – presented in this paper – offers a solution for the facing problem of fast and easy reconfiguration. The hardware of the controller designed modular with software components for online configuration. This solution allows sensor integration on different levels for every part of the manufacturing cell. With unified programming language and the machine specific controllers (post-processing) the cells can be defined easily by different types of human-machine interaction. The shop-floor control architecture is implemented and validated on an Adept SCARA robot. The robot is driven by standalone, low-level, interchangeable, software and hardware components.**

Keywords — robot controller, RT middleware, CNC, SCARA

I. INTRODUCTION

At the previous decades the efficiency of programming increased significantly. In a previous paper more details can be found from the first half of our research [1]. In this paper we describe the latest experiment results and the hardware developments designed since then.

At NC machines the G code is a common language [2]. Over the code we cannot find a universal programming method to configure manufacturing cells. The main concern of the research is to make the possibility for the users to create robot programs more intuitively (e.g. [3, 4]). G code programmed machines have low flexibility and accuracy at manufacturing cells in the view of reconfiguration. This aspect is better at industrial robots. They exhibit high flexibility coupled with a relatively lower absolute accuracy. In some cases this has led to the use of industrial robots as shop-floor controllers [5].

The following technologies and techniques can serve as a basis for overcoming the main challenges at shop-floor controller development:

1) low-cost, accurate and efficient electronics (e.g. microcontrollers) [6]
2) high-speed communication links for inter-machine communication [7]
3) machine programming and development software based on open source solutions (e.g. [8])
4) standards for efficient machine communications and control [8]

The main contribution of this paper is a concept for shop-floor type manufacturing cells, with the opportunity of different level online machine combination. The method gives an opportunity to the common work between several robot, and NC machine types. The next requirements were the most important during the development:
1) The opportunity of common work between different elements of the manufacturing cell independently
2) The communication between the cell elements should be continuous and should have a proper hierarchy.
3) Modular built up for fast hardware modification and cell development.
4) The system should have an easy reconfiguration, and user friendly control interface for the human-robot interaction. The aim of this part is the implementation of the same language on high level for example at the humanoid robot and the conveyor belt.
5) The cell should be able to make high level sensor fusion for feedback during the manufacturing process, and cell modification.

In the view of organization the paper has the following sections: Section I was a short introduction. At section II we can read a short overview about RT-Middleware. Section III presents the brief concept of our shop-floor control method. The experimental results are presented in section IV. Section V concludes the paper.

II. RT-MIDDLEWARE FRAMEWORK

Between industrial robots and NC machines we can find some significant differences in the view of control methods. The NC machines can be used independently for producing parts in small series. The NC machines have a common language (G code). It is an important aspect to follow standards at production. When a part designed, and a G code is generated it must be compatible with every NC machine. At

industrial robots this pressure is absent. Every industrial robot manufacturer can use a brand specific (or sometimes model specific) control language. Despite the standard G code NC machines usually does not have any appropriate port (and sometimes I/O opportunities) for high level communication in a manufacturing cell. Industrial robots are well suited for I/O communication with other machining cell components. Because of the different control language system integrators and shop-floor controllers can be widespread. The mechanics of the NC machines and industrial robots did not changed a lot. The mechanical solutions proved their durability. With the development of the electrical control systems for old or broken machines the robot arms or NC-s can be remained up-to-date. The shop-floor concepts and intelligent cell reconfigurations require sensor fusion for the whole manufacturing cell. This method is also limited by the vendors of the machines and controllers. Due to these issues, many researchers have turned to new and open control architectures for robot systems (so-called middleware systems) [8]. The most important middleware solutions can be found in [10]-[14] and a comparison can be found in [14]. [15]. The main goal of the middleware system is the development of a common language and control concept for different users and tasks. It is also reported that only one of these middleware technologies - called RT (Robot Technology)-middleware - is under standardization [16]. At the beginnings RT-Middleware developed only for humanoid robots. NC machines, industrial robots, humanoids have similar built up at power electronics, mechanics and at low level control methods. RT-middleware has proven to be well adapted to industrial projects and is in use by many different industrial companies (Toshiba, Honda, AIST and also other research institutes) [17]. In 2002, the Japanese Ministry of Economy, Trade and Industry (METI), the Japan Robot Association (JARA) and National Institute of Advanced Industrial Science and Technology (AIST) started a project named "Consolidation of Software Infrastructure for Robot Development". The goal of the project is to implement robot systems in a modular fashion so that they can meet the diverse needs of users. Further, the project aims to allow system designers or integrators to build versatile robots or systems with relative ease by simply combining selected modular parts [16]. In RT-Middleware we can develop our cells with low level components (named RTCs, RTComponents). From RTCs we can create larger operating units like an NC machine. In the lack of an RTC an extra software and/or hardware component can be developed to complete our old machine. In the RT-Middleware environment we can merge, modify, and connect different RTCs easily. The RTCs are platform independent objects. The use of RTCs in shop-floor control architectures has been found to be favorable in other research areas as well [17]. RTCs are the basic, modular units for achieving distributed computing. These elements can be observed as black-boxes. Each black-box has predefined interfaces for communication and the data manipulation and calculation is hidden from the external parties. More details about the RT-middleware can be found in references [17].

III. SHOP-FLOOR CONTROL ARHITECTURE DESIGN

The control of flexible manufacturing systems, until now, was limited to I/O level signaling, using an industrial robot or a PLC as an overall controller. The proposed architecture is based on software components (RT-Middleware) and on standards (STEP family). The shop-floor controller is a normal PC connected to the high-speed communication network of the shop-floor. This PC serves as an interface for system assembly and task specifications. At the bottom level (Cell components) we have all the cell members represented by NC-machines, industrial robots, cameras, sensors, conveyor belts, feeders and etc. In general, all cell members that have the ability to communicate are defined as active cell components. Cell members, if possible or needed, are built up from smaller components, forming controllable objects where the communication drivers between the software components are based on the RT-middleware framework. These components represent building blocks for the manufacturing system and are called RT-Components or RTCs. Depending on the specified task, these RTCs can be used as fully standalone cell members or can be assembled in a way that will satisfy the needs posed by a given task.

IV. EXPERIMENTAL SYSTEM DESCRIPTION AND VALIDATION

Every machine type used in manufacturing cells has different control hardware and software. Because of the standards some components can be the same or very similar. (NC machines, industrial robots, PLC or PC). With a shop-floor concept and RT-Middleware technology the different machines and controllers can be configured and assembled for every given task. For this solution the parts of a cell should have a modular construction for example in the view of axis drives, and sensors. This modification can be hardware or a software solution. (It depends on the given task, and the part of the machine.) At a software problem the part of the cell can be a standalone unit with the high level control and supervision of the system. (In the beginning this is the most convenient) [5] We can implement the software for example on a PC connected to the machine. The PC translates the data between the machine specific language and the RT-Middleware system. In this case the original controller can limit our capabilities with the low level software and the hardware. At a hardware problem the controller of the machine must be exchanged with low-cost highly modularized electrical components [6]. In this case everything has to be developed and implemented from scratch, but as a result we obtain a high level of flexibility and a truly software-based controller. In our experiments the hardware design was an important part of the project to get the most sophisticated hardware and low level software for the support of the system development. Our team could implement and test the components at an Adept 604-S SCARA robot, and a three axis CNC machine. Three of the components was developed on the SCARA robot, and one of the controllers was connected to the CNC machine.

Software and hardware layout [1]

RT-Middleware can be implemented on different levels. For example Linux CNC has an own interpolator, kinematics, etc. modules. For the pre processing hardware controllers RT-Middleware components (hardware and software) must be designed. The following standalone hardware and software components were created (RTCs):

a) *Programming interface:* this is a PC-based graphical user interface. The Linux CNC uses also the same method for robot programming. We can enter coordinates, G-code, etc.

b) *Interpolator:* it plans the trajectory of the robot with the knowledge of the actual position and orientation of the TCP and the workspace. The software calculates the step positions of the planned trajectory in Cartesian coordinate system.

c) *Inverse Kinematics:* the module calculates the joint coordinates of a machine from the next position, orientation, and velocity of the TDP. The inverse kinematics is different for every type of robot. The equations must be implemented individually for every machine. The frame of the component, and the input, output parameters are the same.

d) *Direct Kinematics:* the module calculates the position and orientation of the TCP from the joint angles and velocities. Similar to the inverse kinematics in this module the equations of the motion are also different for every machine types, but the frame of the program is always the same.

e) *Joint Controller:* in some cases the controller part of a machine is only one joint. The module gives speed, position, or torque references for the joint.

The most important implementation was the scara kinematics during the experiments, measurements and hardware development. (Fig. IV.1.)
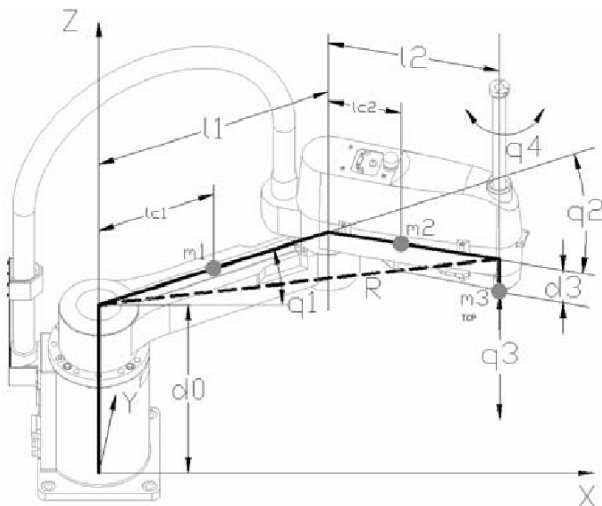

Fig. IV.1. Mechanical drawing of the Adept 604-S SCARA robot. *m1, m2, m3* are the masses, *l1,l2,d0,d3* are the length, *q1,q2,q3,q4* are the angles of the corresponding joints. (These data are necessary only for the calculations of robot dynamics: *lc1* and *lc2* are the masses position on joint 1 and joint 2, respectively.)

The Cartesian coordinates of TCP can be expressed as equation (4.1), (4.2), (4.3).

$$x = l_1 \cdot \cos q_1 + l_2 \cdot \cos(q_1 + q_2) \qquad (4.1)$$

$$y = l_1 \cdot \sin q_1 + l_2 \sin(q_1 + q_2) \qquad (4.2)$$

$$\rho = q_1 + q_2 + q_4 \qquad (4.3)$$

Forward kinematics can be expressed deriving equations (4.1), (4.2) as (4.4),(4.5).

$$\dot{x} = -l_2 \cdot \sin(q_1 + q_2) \cdot (\dot{q}_1 + \dot{q}_2) -$$
$$- l_1 \cdot \sin q_1 \cdot \dot{q}_1 \qquad (4.4)$$

$$\dot{y} = l_2 \cos(q_1 + q_2) \cdot (\dot{q}_1 + \dot{q}_2) +$$
$$+ l_1 \cdot \cos q_1 \cdot \dot{q}_1 \qquad (4.5)$$

The connection between the Cartesian velocities and joint velocities can be represented as equation (4.6).

$$\dot{X} = \underline{\underline{J}}(q) \cdot \dot{\Theta} \qquad (4.6)$$

Where
$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$$

$$\dot{\Theta} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

Where $J(q)$ is a Jacobean matrix as equation (4.7)

$$J(q) = \begin{bmatrix} -l_1 \cdot \sin q_1 - l_2 \cdot \sin(q_1 + q_2) & -l_2 \cdot \sin(q_1 + q_2) \\ l_1 \cdot \cos q_1 + l_2 \cdot \cos(q_1 + q_2) & l_2 \cdot \cos(q_1 + q_2) \end{bmatrix} \quad (4.7)$$

Technical solutions

In this part we present four solutions for different level control methods at a manufacturing cell.

The first one is a joint controller. (Fig. IV.2.) Two RT-Middleware based joints were developed for the first two joints of Adept SCARA 604-s robot. The cores of both are SUZAKU boards from Atmark Techno Inc, which are embedded device platforms based on the combination of FPGA and Linux. The following two chapters describe SUZAKU-V and SUZAKU-S platforms. The board can get the references via Ethernet, and give the calculated references to the servo amplifier via serial port. The system can calculate position, speed, and current (torque) control for brushed DC motors. This is the smallest element of a manufacturing cell.
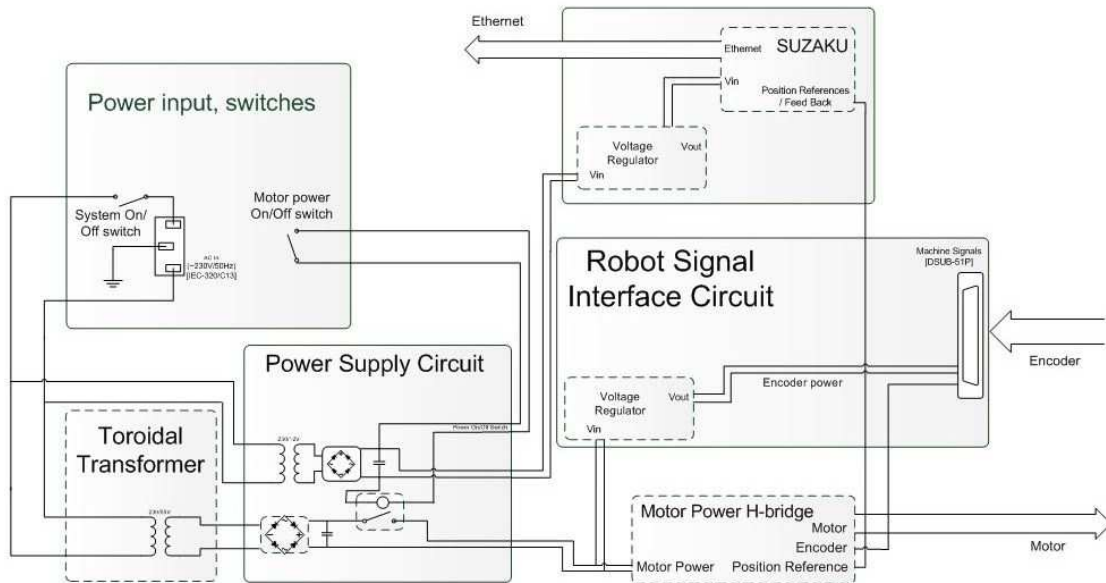
Fig. IV.2. The block diagram of the joint controller RTC

The second one is a simple decentralist CNC controller.
(Fig.IV.3) Most of the controllers are standalone units and are able to move the TCP without a computer. External computers are only used for offline programming. This means that the program is written on a PC without the robot, and after that the user can upload it to the robot controller's memory for running it. Our controller is able to work with a PC based interpolator. The external interpolator is generally a PC based real-time application. It is possible to control additional functions along output 8 relays, and 8 isolated inputs, for example coolant, tool size measurements, and other PLC functions. These I/O interfaces can be accessed along ModBus interface.
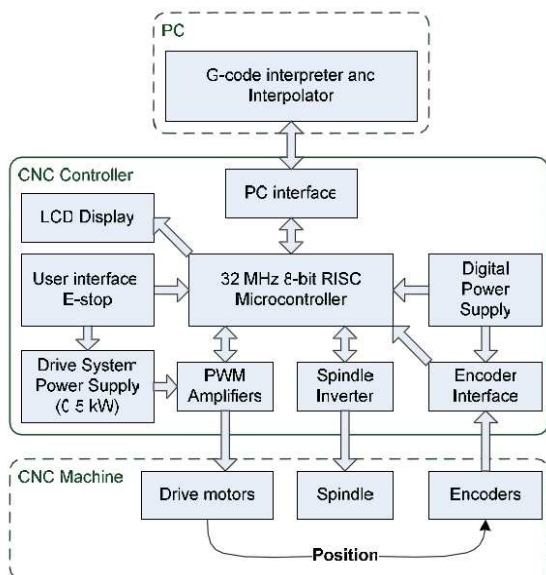

Fig.IV.3. The block diagram of the CNC controller

The third one is the first version of a universal robot controller.
(Fig.IV.4.) For handling user interface and computing independent PID loops a simple 8-bit RISC microcontroller is eligible.

A controller like this can simply be universal, because there are no machine specific computing processes that have to be customized for different mechanical systems. For independent joints, there are many simple adaptive control methods, for learning the mechanical parameters. Hence with suitable power stages, and a versatile encoder and I/O interface, the equipment is able to drive different machines with a short configuration before.

To control a nonlinear dynamic system the computing hardware has to solve the inverse dynamic Lagrange equation. This FPGA-DSP platform is able to compute all the functions that a standalone unit needs: Interpolation, path planning, inverse kinematics and inverse dynamics calculations, encoder feedback count, PID (or PD) control with computed torque method, data conversions, and external interfaces to PC and other industrial units.

To make this architecture suitable for different robots, that is not enough to design versatile power stages and I/O interfaces. The robot's dynamic model has to be integrated into the firmware. That's make this problem difficulty, and there is no prevalent method today to solve within a short time and without a specialist. All the two different controllers have an Open RT-Middleware interface for PC based interpolator. This makes a controller able to work with other controllers along a real time common interpolation, which have to be created in the RT-Middleware framework for the custom application.
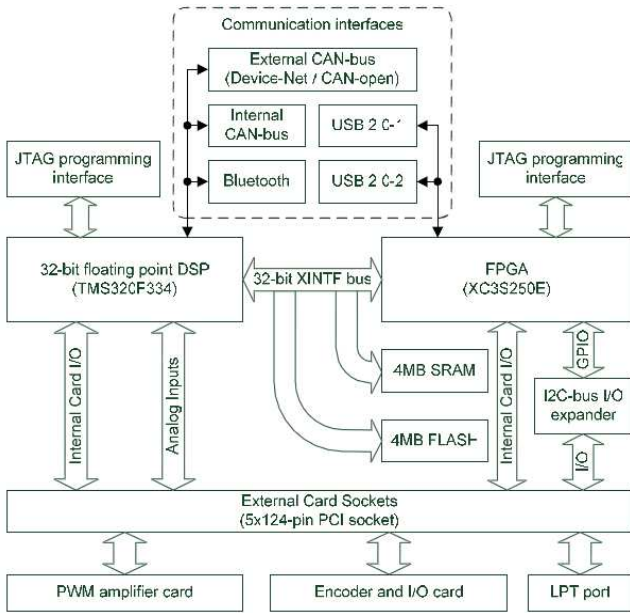
Fig.IV.4. The block diagram of the nonlinear motion controller

The fourth one is the second version of the universal robot controller. (Fig.IV.5.) After a PCI bridge interface application specific IC there is an FPGA on a self designed PCI card, therefore it is possible to handle reference and feedback signals. Servo controllers, encoders, end switches etc. can be connected to the computer. We designed the system around the Linux CNC, what is an open source robot controller software with ladder logic editor, PLC functions, G code editor, etc. so the controller can handle other peripherals in line with the robot, for example tool library, assembly line etc. Linux CNC also has a modular built up. The RT-Middleware module for Linux CNC is under development.
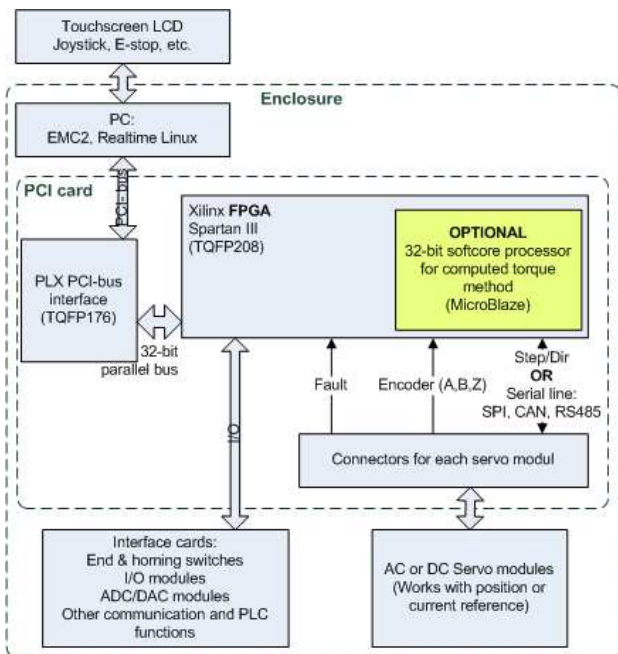

Fig.IV.5. The block diagram of the universal robot controller

Experimental results

At the example controllers at the different levels we made measurements to the validation of the hardware, and the control architecture. At most of the measurements we tested the system on the major joints of the Adept SCARA robot. A robust, decentralized PID control was implemented in the Joint Controller (First level controller). At the CNC controller (second level controller) we implemented three PID controllers. At the first version robot controller (Third level controller) we implemented PIC controllers for every joint with a DSP calculated interpolator. At the second version robot controller (Fourth level controller) we implemented PID controllers for every joint. (Kinematics and other Linux CNC modules are also implemented.) At the Linux CNC we could easily measure every parameters of the robot at the same time. (For example joint coordinates TCP position, orientation, feedback, reference, etc.). In the view of the other controllers we could measure the references, the error, the current, and some other parameters of a DC motor. (Fig.IV.6.) In this case the controller of the power electronics measured the data at 10 kHz frequency. At the Linux CNC measurement the sampling time was 100 kHz, (Data measured by the PCI card.)
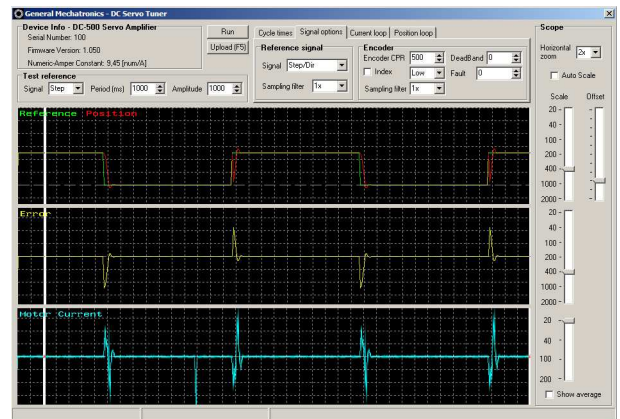

Fig.IV.6. The user interface of the tuner program

V. CONCLUSION

This paper presented the basic concept of shop-floor control architectures and four examples for different level machine controllers based on RT-Middleware framework and STEP-NC standard. The time demanding manufacturing cell reconfiguration requires new programming methods and machine-user interactions. The RT-Middleware technology with the available standards can give a solution for fast and easy manufacturing cell configuration. The controllers from the different levels are implemented and tested on the joints of a SCARA robot and on a 3D milling machine. The experimental results shows no significant drawbacks of using a networked, software component based architecture. The next step will be the test and development of the second version universal robot controller concept. (Fourth level controller with the Linux CNC and PCI card conception. (Fig.IV.6.))

References

[1] Bjorn Solvang, Gábor Sziebig, Péter Korondi, "Shop-floor arhitecture for effective human-machine and inter-machine interacion" in Acta Polytechnica Hungarica of 2012., Budapest, Vol.9 No.1. pp. 183-201, 2012

[2] S.T. Newman, A. Nassehi, X.W. Xu, R.S.U. Rosso Jr., L. Wang, Y. Yusof, L. Ali, R. Liu, L.Y. Zheng, S. Kumar, P. Vichare and V. Dhokia, Strategic advantages of interoperability for global manufacturing using CNC technology, Robotics and Computer-Integrated Manufacturing, Vol. 24, No. 6, pp. 699-708, Dec. 2008.

[3] J.N. Pires, G. Veiga and R. Araujo, Programming-by-demonstration in the coworker scenario for SMEs, Industrial Robot, International Journal, Vol.36,No.1, pp. 73-83, 2009.

[4] S. Ekvall and D. Kragic, Robot Learning from Demonstration: A Task-level Planning Approach, International Journal of Advanced Robotics, Vol.5, No.3, pp.223-234, 2008.

[5] Bjorn Solvang, Gábor Sziebig, Péter Korondi, Shop-floor arhitecture for effective human-maschine interaction, Acta Polytechnica Hungarica Vol.9, pp. 107-120, 2012

[6] J.U. Cho, Q.N. Le and J.W. Jeon, An FPGA-Based Multiple-Axis Motion Control Chip, IEEE Trans. Ind. Electron., Vol. 56, No. 3, pp. 856-870, Mar. 2009.

[7] T. Li and Y. Fujimoto, Control system with high-speed and real-time communication links, IEEE Trans. Ind. Electron., Vol. 55, No. 4, pp. 1548-1557, Apr. 2008.

[8] Bence Kovács, Géza Szayer, Ferenc Tajti, Solvang Bjorn, Péter Korondi, Design of a universal robot controller Robi, International Engineering Symposium at Bánki (IESB 2011). Budapest, Hungary, pp. 1-13, 2011.11.15.

[9] International Organization for Standardization (ISO), ISO 14649-1, Industrial automation systems and integration – Physical Device Control – Data model for computerized numerical controllers Part 1: Overview and fundamental principles, 2003.

[10] Ozaki Fumio, Oaki Junji, Hashimoto Hideaki, Sato Hirokazu, "Open Robot Controller Architecture (ORCA)" in Journal: Nippon Kikai Gakkai Robotikusu, Mekatoronikusu Koenkai Koen Ronbunshu, Vol. 2, 2003.

[11] M. Mizukawa, H. Matsuka, T. Koyama, T. Inukai, A. Noda, H. Tezuka, Y. Noguchi, N. Otera, "ORiN: open robot interface for the network - the standard and unified network interface for industrial robot applications" in Proceedings of the 41st SICE Annual Conference (SICE 2002), Tokyo, Japan, Vol. 2, pp. 925- 928, Aug. 2002.

[12] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, Yoon Woo-Keun, "RT-middleware: distributed component middleware for RT (robot technology)" in Proceedings of 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 3933- 3938, ISBN: 0-7803-8912-3, 2-6 Aug. 2005.

[13] G. Veiga, J. N. Pires,K. Nilsson "On the use of SOA platforms for industrial robotic cells" in Proceedings of Intelligent Manufacturing Systems (IMS2007), Spain, 2007.

[14] J.-C. Baillie, "URBI: towards a universal robotic low-level programming language" in Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 820- 825, ISBN: 0-7803-8912-3, 2-6 Aug. 2005.

[15] B. Solvang, G. Sziebig, and P. Korondi, "Multilevel control of flexible manufacturing systems," in Proc. IEEE Conference on Human System Interactions (HSI'08), pp. 785– 790, 2008.

[16] G. Sziebig, A. Gaudia, P. Korondi, N. Ando, B. Solvang, "Robot Vision for RT middleware Framework," in Proc. IEEE Instrumentation and Measurement Technology Conference, pp. 1-6, 2007.

[17] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, Yoon Woo-Keun, "RT-middleware: distributed component middleware for RT (robot technology)" in Proceedings of 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 3933- 3938, ISBN: 0-7803-8912-3, 2-6 Aug. 2005.