Recent Innovations in Mechatronics (RIiM) Vol. 1. (2014). No. 1-2.
DOI: 10.17667/riim.2014.1-2/2.

DEVELOPMENT OF A MICROCONTROLLER-BASED SYSTEM WITH HARDWARE-IN-THE-LOOP METHOD FOR REAL-TIME CONTROL ALGORITHMS PROTOTYPING

Dr. Csaba SZÁSZ
Department of Electrical Machines and Drives
Faculty of Electrical Engineering, Technical University of Cluj
Cluj, Romania
Csaba.Szasz@emd.utcluj.ro

Zoltán FÜLÖP
Department of Electrical Machines and Drives
Faculty of Electrical Engineering, Technical University of Cluj
Cluj, Romania

***Abstract*— It is well known that the hardware-in-the-loop simulation (HILS) is a wide range used method for control prototyping and system modeling applications. This paper presents a microcontroller-based control system design and development for real-time control algorithms prototyping using the HILS strategy. The entire hardware module of this system is based on a BigPIC6 development board interfaced to the Matlab/Simulink software environment. The plant is modeled by using Simulink components on the xPC target and the control algorithms are implemented on the microcontroller-based module. As a result of the adopted development strategy, a powerful and versatile system has been obtained for real-time control algorithms prototyping and development. A large set of experimental results proves the feasibility and efficiency of the developed HILS system, providing a very efficient research and development tool both for modeling and experimental purposes.**

Keywords—component; formatting; style; styling; insert (key words)

I. INTRODUCTION

Generally speaking, international references define HILS as a synthesis paradigm used in the development and test of complex real-time embedded systems combining many advantages both of physical and virtual prototyping. The essence of such a definition is that it is a technique which combines the mathematical simulation model of a system (or plant) with actual physical hardware, by capturing closed-loop bidirectional interactions between its physical and virtual constituents [1, 2]. In other words, in a modeled system the hardware performs as through it were integrated into the real control system. However, it is no doubt that modern engineering systems require a high degree of prototyping and HILS has become indispensable in many industrial applications, such as aerospace, industrial automation, power plants, nuclear plants, automotive systems, marine, military applications, etc.

There the first major advantage of this method is considered its cost effectiveness, in the vast majority of applications requiring significantly less hardware then fully physical prototypes. HILS also outstands by its rapid prototyping facilities because they can be considered quicker to build and tested in the loop by utilizing the adequate software toolkits [3, 4]. At the same time, they offer a high degree of repeatability, comprehensiveness (HILS often makes possible experiments in a wide range of operating conditions than what is feasible via physical prototyping), and a high safety of testing critical industrial processes or safety-critical applications (high speed train systems, supersonic aircraft, nuclear plant processes, etc.).

Additionally, HILS remarks in many applications with higher processing speed by enabling concurrent systems engineering as well [3, 4, 5]. Not at least, HILS outstands through its non-destructive nature, often making possible to simulate a wide range of destructive or irreversible events, avoiding costly experiments and tests.

Considered the above expressed theoretical remarks, it is not difficult to assume that the HILS method offers a great amount of possible solutions for a wide range of control applications implementation and development. Among these one of the most important applications of this strategy is in the controller design and testing, respectively in the real-time control algorithms prototyping and development. There hardware control units performing physical tasks are connected with virtual devices and virtual models of systems being controlled. Therefore, real hardware modules are integrated with virtual models emulating realistic operating conditions based on high-fidelity systems. In this reason, the major aim of this paper is to present the design and development steps of such a controller-in-the-loop simulation system and to outline its advantages, performances, and versatility. The final goal is to experiment a powerful HILS system for real-time control algorithms rapid prototyping.

II. THE MICROCONTROLLER-BASED HARDWARE-IN-THE-LOOP SIMULATION SYSTEM STRUCTURE

The theoretical background of the development efforts regarding the implementation of the microcontroller-based control system with hardware-in-the-loop (HIL) method has been built upon the classical closed-loop control theory main concepts.

According to this, it is considered the block diagram of a closed-loop system, where the output magnitude is controlled by imposing a reference magnitude to the system input, which is compared them with the measured one and the resulting error is processed by a controller being able to generate the adequate control signal to the plant input. In the first step of the experiments the endeavor is to break the control loop and to acquire the error signal by a microcontroller-based hardware development system. Then the captured signal will be real-time processed, according to the algorithms and rules implemented on the digital controller unit. As shown below in figure 1, the loop will be closed again by connecting the digital controller output to the plant input and providing the novel control signal.

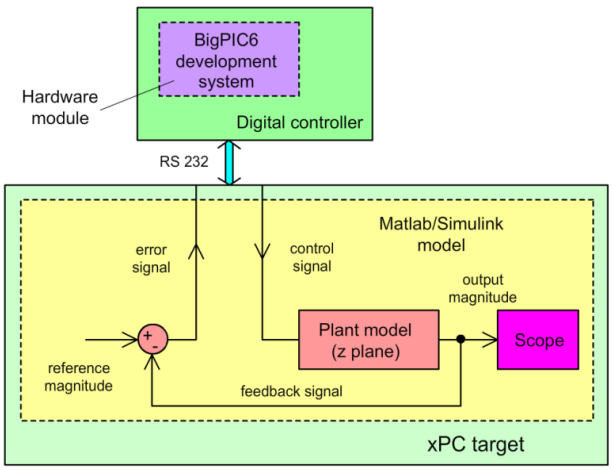

Fig. 1. The block diagram of the microcontroller-based HILS system

It is not difficult to observe from the block diagram that the Matlab/Simulink model of the controlled process is connected to an actual physical hardware, by capturing closed-loop bidirectional interactions (RS 232 communication) between physical and virtual constituents. Therefore, the BigPIC6 development system-based digital controller, embedding a Microchip PIC-type central processing unit it is integrated with a software model and will perform realistic operating conditions, by processing real-time control algorithms. It is self-understanding that in such a digital control system the plant model is expressed in the z plane, and the numerical simulation is running on xPC target. The major advantage of this system configuration is that allows a great freedom for the software developers for rapid real-time control algorithms prototyping. In this way, a large set of various control strategies (classical PID, fractional controllers, fuzzy algorithms, robust control, etc.) becomes possible to be designed, implemented, and tested by using the HILS method.

III. THE HILS SYSTEM DEVELOPMENT AND IMPLEMENTATION

As it has been mentioned before, the main goal of the research is to develop a powerful and versatile system for real-time control algorithms prototyping using the HILS strategy. Therefore, instead of analyzing various control system architectures to reach the most adequate output magnitudes for a desired control plant or to evaluate systems dynamic response behaviors, there for an arbitrarily given plant a hardware framework for control algorithms testing and experimenting it is expected to be developed. In other words, there the focus is on the digital controller part, which must be powerful enough to real-time compute a large set of various control algorithms, in order to implement the most performing program or control task for a considered application. For this purpose the microcontroller-based BigPIC6 hardware development system given in figure 2 is considered [6].

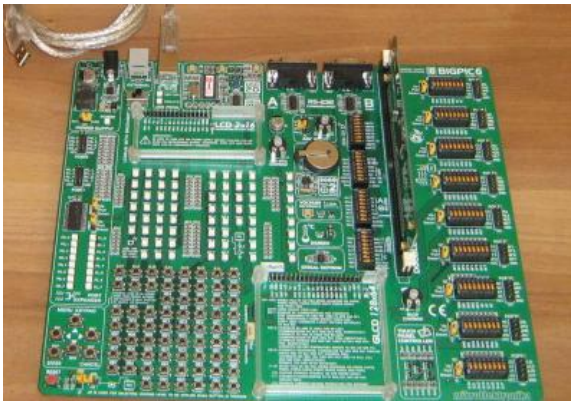

Fig. 2. The BigPIC6 hardware development system

This is a powerful and versatile development tool for programming and experimenting with Microchip PIC18Fxx controllers, including on-board programmer with *mikroICD* support providing an interface between the microcontroller and a personal computer. The central processing unit of this system (a microcontroller from the PIC18Fxx family) is rich interfaced with a large set of on-board hardware modules, such as alphanumeric 2x16 LCD display, 128x64 graphic LCD display, or an I/O port expander, allowing easy to simulate operation of the target device.

The BigPIC6 also provides an USB 2.0 programmer, two RS-232 ports (RS-232A and RS-232B) for serial communication, serial EPROM using I^2C communication, a 10 bit A/D converter, DS1829 temperature sensor, real-time

clock, 67 LED's to indicate the input/output pins' logic state, a keyboard with 67 push buttons, a navigation menu keypad, and a set of nine 10-pin connectors connected to the microcontroller I/O ports [6,7].

The enumerated rich hardware resources allow a sufficient freedom to the designer to develop and prototype a large set of real-time control algorithms for a given application. Therefore, the classical process control-loop block diagram will be broken-up by interfacing the PIC microcontroller-based development system, as it has been expressed in figure 1. It means that the entire BigPIC6 development system will play now the full role of a digital controller uploaded with various control tasks. In this way a software model developed on the *xPC* target will be linked with a physical module capturing closed-loop bidirectional interactions by RS-232 communication facilities. It means that a mathematical simulation model of a system (or plant) it is combined with a hardware platform, by closing the so-called "hardware-in-the-loop" simulation structure.

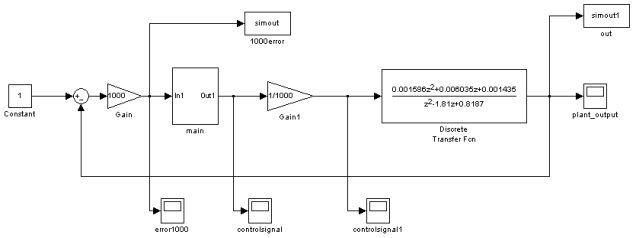

Fig. 3. The Simulink model of the hardware-in-the-loop simulation system

In order to complete the above mentioned theoretical remarks, it is considered an arbitrarily chosen plant with its function transfer expressed in the z plane as follows:

$$H_p(z) = \frac{0.001586 \cdot z^2 + 0.006035 \cdot z + 0.001435}{z^2 - 1.81 \cdot z + 0.8187}. \quad (1)$$

This plant has been included then into a Matlab/Simulink model-based closed-loop control system as shown in figure 3. There the *main* Simulink block with its *In1* and *Out1* input/output signals represents the communication interface between the mathematical model and the BigPIC6-based digital controller. During the entire experiments the transfer function of the plant will remain unchanged, being modified only the reference signal set in the block named *Constant*.

One of the most important challenges regarding the implementation of the HILS system is linked with the bidirectional communication establishment between the Matlab/Simulink software environment and the microcontroller-based BigPIC6 development system, respectively the data passing between these software and hardware constituents. For this reason a specially designed RS-232 communication-based algorithm has been implemented and tested by interconnecting the *xPC* serial port with the BigPIC6's RS-232A port. The essence of this strategy is that a floating-point signed real value computed in the Simulink model (namely the error signal value) will be ASCII encoded and transmitted via the *xPC* computers' serial communication port. On the receiver side this information it is captured by using controllers' RS-232A port and reconverted into the initial signed real value. This information it is processed then according to the real-time control algorithm implemented in a *mikroPascal* code in the digital controller-side. The result obtained (namely the control signal value) it is encoded once again into an ASCII code packet and returned via serial communication to the Simulink model. There the captured information will be decoded by using the algorithm presented in figure 4. The floating-point signed real value obtained from this algorithm represents in fact the input magnitude for the plant and in this way the control loop it is closed once again.

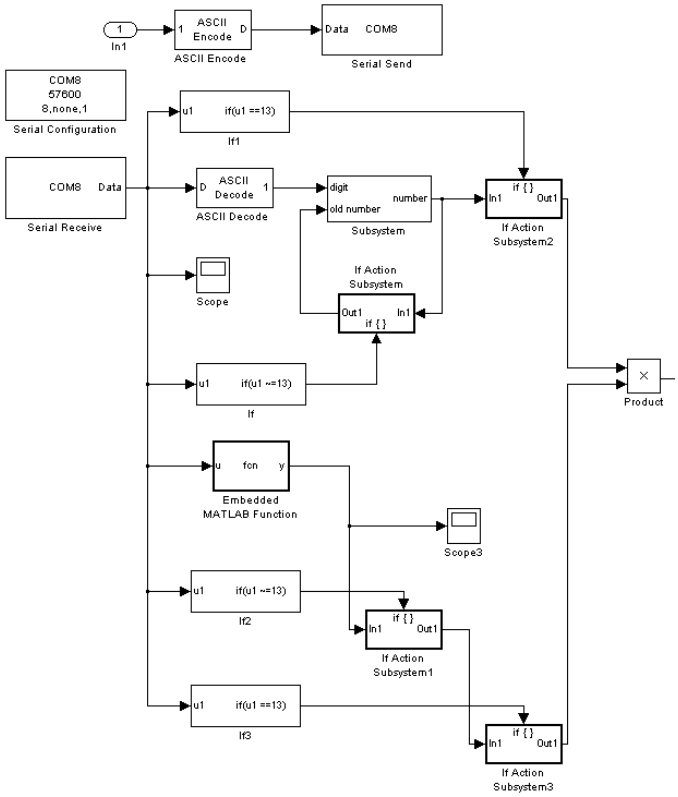

Fig. 4. Data passing via the Matlab/Simulink model and the BigPIC6-based digital controller

Once the problem of the real-time data passing between the Matlab/Simulink software environment and the microcontroller-based hardware controller has been solved, careful test operations regarding the HILS system operation can be started. In figure 5 it is shown the laboratory setup implemented for this purpose. There in the right side of the figure it can be identified the BigPIC6 development system as the digital controller being integrated into the Matlab/Simulink

simulation environment. This is interfaced via its RS-232A serial port to the personal computer's serial communication.

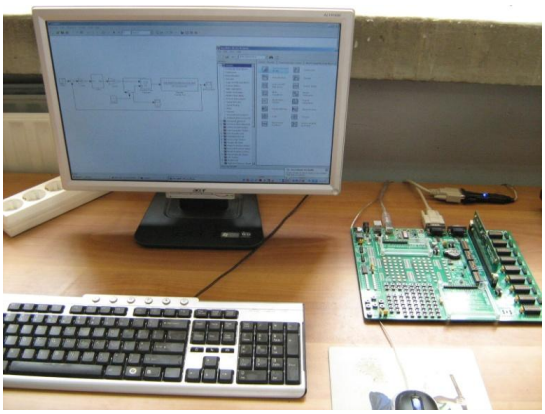

Fig. 5. The laboratory setup HILS system

On the xPC target's monitor can be followed the running Simulink model which passes the computed floating-point signed values via its serial port to the digital controller. This hardware unit decodes the captured information, processes it according to the implemented algorithm, and then returns the result to the personal computer. This will be a floating-point signed value introduced as an input control signal to the plant given in the Simulink model. The hardware resources and processing speed of the PIC-type microcontroller are full enough to ensure a real-time communication and hardware-in-the-loop simulation process.

IV. EXPERIMENTS AND HIL SIMULATION RESULTS

The entire HIL simulation and experimentation process follow a simple idea. In the first step, the theoretical study by Matlab/Simulink simulation of the given plant in *z* plane it is performed. There for a given set of the simulation parameters the dynamic response of the system is measured and evaluated. Next, for the same parameters and conditions a HIL simulation of the system is performed by using the experimental setup presented in figure 5. If the obtained results for the two different simulation strategies are near the same, then it can be considered that the experimental results covers the theoretical ones, and the HILS system works properly. Therefore, it can be considered that the entire system is ready to be used for various real-time algorithms implementation and prototyping. In order to do not excessively complicate the experimenting efforts, in the first stage of the measurements a simple *P*-type (proportional) controller algorithm has been implemented on the microcontroller-based BigPIC6 development system. Then, for a given input reference signal the system response is analyzed, by considering a set of proportional constants programmed in the digital controller. Following the above outlined strategy, in figure 7 it is presented the first result plotting the system response corresponding to the output magnitude indicated in figure 3, for a given set of simulation parameters. There the following constants have been considered: the input reference level is $i_R=1$, the proportional coefficient $k_P=3.1$, the sampling period is $T_s=0.0025s$, and the simulation time is $t=10s$. Obviously, this is a result obtained in pure theoretical condition, with a fast system response, low overshot, and without steady-state error.

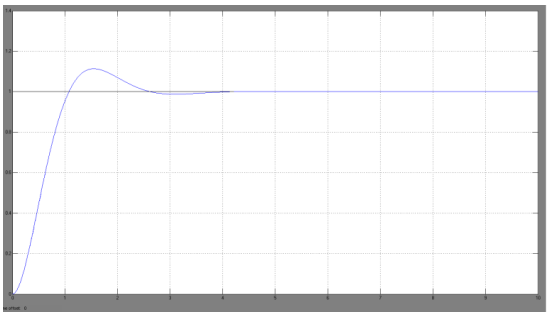

Fig. 6. The system response in Matlab/Simulink simulation (z plane, $i_R=1$, $k_P=3.1$, $T_s=0.0025s$)

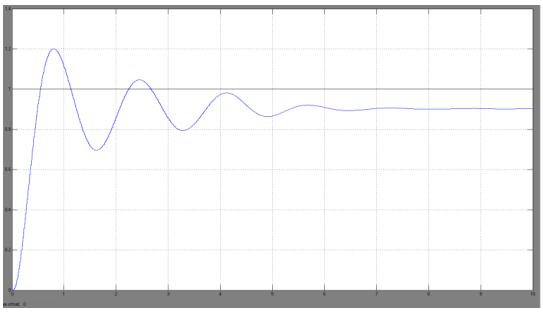

Fig. 7. The system response in HILS ($i_R=1$, $k_P=3.1$, $T_s=0.0025s$)

In the next step a HILS has been performed, in the same simulation conditions and with the same parameters set (figure 7). In this second case it is not difficult to observe, that the experimental result do not cover the theoretical ones. There significant oscillations of the output magnitude are remarked, and the steady-state error also is significant (around 10% of the input reference level).

The above presented results shows that in order to achieve better HILS dynamic responses, a much closely correlation between the hardware resources of the microcontroller-based BigPIC6 controller and the Matlab/Simulink simulation environment parameters it is necessary to be chosen. In the hardware component side, the processing speed it is determined mainly by the microcontroller unit's (MCU) 8 bit resolution, the system clock frequency (20MHz), respectively by the programmed RS-232 communication speed. Obviously, these parameters can't be changed therefore it remains as a

reliable solution to optimize the implemented controller algorithm processing speed or to tune more appropriate simulation parameters as well. In this reason, a next set of simulation results shows that if the proportional constant k_p of the controller is decreased, better results can be achieved.

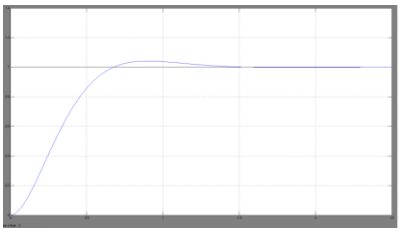

Fig. 8. The system response in Matlab/Simulink simulation (i_R=1, k_P=2, T_s=0.0025s)

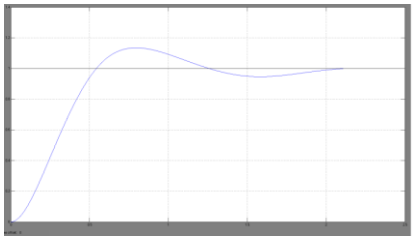

Fig. 9. The system response in HILS (i_R=1, k_P=2, T_s=0.0025s)

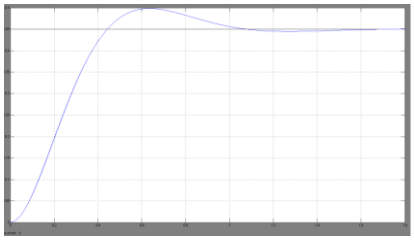

Fig. 10. The system response in Matlab/Simulink simulation (i_R=0.45, k_P=3, T_s=0.0025s)

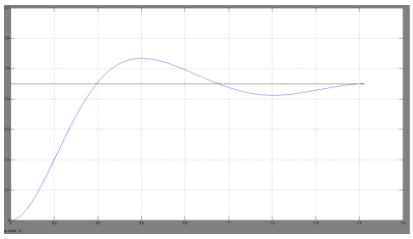

Fig. 11. The system response in HILS (i_R=0.45, k_P=3, T_s=0.0025s)

system simulation parameters cross some limits, the microcontroller-based BigPIC6 digital controller will do not have sufficient hardware resources to maintain the system response inside of the desired ranges where it can be surely

In figures 8 and 9 are presented the theoretical, respectively the HIL simulation results using the following system parameters: i_R=1, k_P=2, T_s=0.0025s. It can be observed, that here the proportional constant has been decreased to the level of 2.0 units. As a result, the HIL simulation result covers fairly the theoretical ones, the system dynamic response is adequate and the steady-state error becomes zero. Only the oscillation ripples are significantly higher than in the theoretical simulation case obtained. More acceptable results are given in figures 10 and 11. There the input reference signal level has been decreased to the i_R=0.45 levels and the experimental result looks nearly the same as in the case of the theoretical approach.

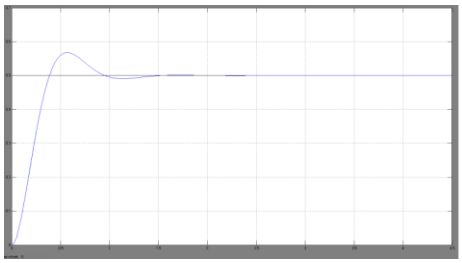

Fig. 12. The system response in Matlab/Simulink simulation (i_R=0.5, k_P=3.5, T_s=0.0025s)

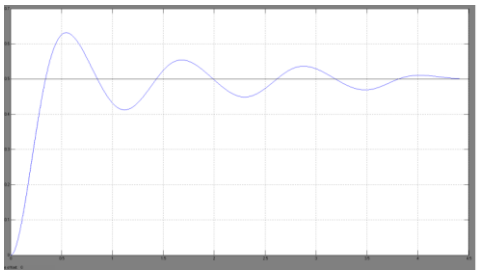

Fig. 13. The system response in HILS (i_R=0.5, k_P=3.5, T_s=0.0025s)

Even for a lower reference input signal level, if the proportional constant is increased to higher levels (k_P=3.5 in figures 12 and 13), the HIL simulation shows higher output signal oscillation ripples.

The above ranked results prove that the endeavor to choose the adequate simulation parameters set is essentially to obtain the desired HILS results. By carefully tuning the optimal values, it is possible to reach a parameters bandwidth where the experimental simulation results overlap the theoretical ones. In such situation it can be stated that the HILS system works proper and it can be used for real-time algorithm prototyping. On the other side, also it is very clear that if the declared that the experiments covers the theoretical approach. In such situations the HILS system it cannot be used for algorithm rapid prototyping and development, because the

results will be significantly distorted and far from the expectations.

V. CONCLUSION

The paper presents promising theoretical and experimental results regarding a high performance HILS system development and experimentation. The proposed solution relies on a configuration which combines the mathematical simulation model of or plant with actual physical hardware, by capturing closed-loop bidirectional interactions between its physical and virtual constituents. There the chosen architecture based on BigPIC6 development system playing the role of the digital controller looks a feasible, cheap and versatile solution for a high-performance real-time algorithms prototyping platform development. The obtained laboratory experiments underline that by using this configuration it is possible to reach adequate results to develop a powerful HILS system for a large set of control algorithms implementation. At the same time, future works must be concerned for tuning better simulations parameters set in order to reach a more closely correlation between the hardware resources of the microcontroller-based BigPIC6 controller and the Matlab/Simulink simulation environment behaviors.

REFERENCES

[1] H. K. Fathy, Z. Filipi, J. Hagena, J. Stein, *Review of Hardware-in-the-Loop Simulation and its Prospects in the Automotive Area*, Modeling and Simulation for Military Applications, proc. of SPIE 6228, doi: 10.1117/12.667794 (2006).

[2] R. Isermann, j. Schaffnit and S. Sinsel, *Hardware-in-the-Loop Simulation for the Design and Testing of Enginee-Control Systems*, Control Engineering Practice, vol. 7. Pp. 643-653, (1999).

[3] W. Grega, *Hardware-in-the-Loop Simulation and its Application in Control Education*, Proceedings of the Frontiers of Education Conference, vol. 2, pp. 12-6, (1999).

[4] N. Noomwongs, H. Yoshida, M. Nagai, k. Kobayashi and T. Yokoi, *Study on Handling and Stability using Tire Hardware-in-the-Loop Simulator*, JSAE Review, vol. 24. Pp. 457-464, (2003).M. Gomez,

[5] Hardware-in-the-Loop Simulation, http://www.embedded.com/story/OEG20011129S0054

[6] MikroprogSuite for PIC – BigPIC6 Development system, Mikroelectronika, www.mikroe.com.

[7] G Husi;P T Szemes; E Dávid; T I Erdei; G Pető :Reconfigurable Simulation and Research Toolset for Building Mechatronics Proceedings of CERiS'13 - Workshop on Cognitive and Eto-Robotics in iSpace. Budapest, Hungary, 2013.03.19-2013.03.20.BME, 2013. pp. 26-31.