Raspberry Pi and Arduino – a perfect couple for control education

Jaroslav Sobota * **Roman Pišl** ** **Pavel Balda** ***
Miloš Schlegel ****

* *European Centre of Excellence*
NTIS - New Technologies for Information Society
Faculty of Applied Sciences, University of West Bohemia in Pilsen
Univerzitní 8, 301 00 Plzeň, Czech Republic
(e-mail: jsobota@kky.zcu.cz)
** *(e-mail: rpisl@kky.zcu.cz)*
*** *(e-mail: pbalda@kky.zcu.cz)*
**** *(e-mail: schlegel@kky.zcu.cz)*

control education; control equipment; real-time; closed-loop control

Abstract: Nowadays the control education usually heavily relies on the available simulation packages and virtual laboratories, both of which have their irreplaceable position in the educational process. But unfortunately the closed loop experiments are way too often limited to these virtual domains and the students lack the physical feedback about the impact of control algorithms and its parameters. The reasons might be the price of the hardware setup to demonstrate the control algorithms physically or a complicated transition from simulation to real-time platform. This paper describes an extremely inexpensive, straightforward and surprisingly powerful platform for implementation of real-time control algorithms. The platform consists of an Arduino board and a Raspberry Pi running the REX Control System. The Arduino board is used for interaction with the physical world via its inputs and outputs. The REX Control System allows the students to develop and verify the control algorithms in Simulink and then run it in real-time by a few mouse clicks. However, the REX Control System is by no means dependent on Simulink, it is fully functional even if Simulink license is not available. The platform further bridges the gap between the virtual and physical worlds as it is tightly connected to PIDlab.com and Contlab.eu portals, which makes it an ideal choice for control education purposes.

1. INTRODUCTION

Nowadays the control education usually heavily relies on the available simulation packages and virtual laboratories, both of which have their irreplaceable position in the educational process. But unfortunately the closed loop experiments are way too often limited to these virtual domains and the students lack the physical feedback about the impact of control algorithms and its parameters. The reasons might be the price of the hardware setup to demonstrate the control algorithms physically or a complicated transition from simulation to real-time platform.

This paper describes an extremely inexpensive platform, which solves all the above mentioned problems. The platform was developed at a university, initially for internal use, but it soon found its way to other universities, high schools and even hobbyist and DIY communities, which are natural (or rather essential) part of the Arduino [1] and Raspberry Pi ecosystems [2] . The mentioned microcomputers, which the platform is based on, form a perfect couple for control education, combining the computational power, onboard memory and Ethernet connectivity of the Raspberry Pi and relatively rich input-output capabilities of the Arduino board.

Integral part and the main advantage of the presented platform is the software, which turns these two boards into an industrial programmable controller (at least from the educational point of view) as it allows the students/users to create control algorithms using the same software tools and workflow concepts, which are used when developing control algorithms for true industrial controllers and control systems.

Although the current versions of MATLAB/Simulink or LabVIEW support the Arduino and Raspberry Pi boards to some extent, the platform presented in this paper offers additional functionality and wider input possibilities.

2. STRUCTURE OF THE PLATFORM

The educational control platform consists of an Arduino microcontroller board and a Raspberry Pi computer with the runtime of the REX Control System installed [Balda et al., 2005]. The platform is known as REXduino and it is depicted in Fig. 1. The individual components are discussed in the following sections.

[1] Arduino is a trademark of Arduino Team.
[2] Raspberry Pi is a registered trademark of the Raspberry Pi Foundation.

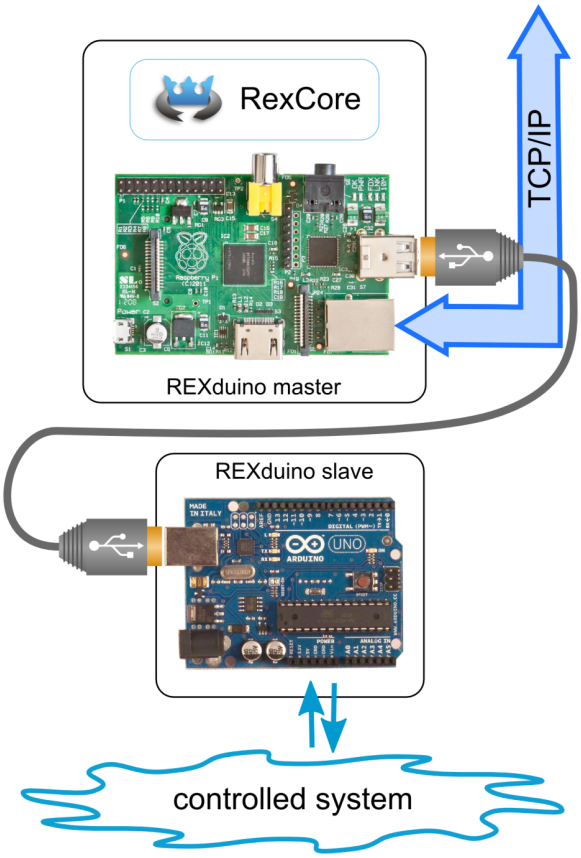

Fig. 1. The REXduino educational platform

2.1 Arduino board

The Arduino is a well-known open-source electronics prototyping platform, which is a central control unit of many embedded control applications ranging from interactive clothing to robotics and 3D printers. In our case the board is programmed to act as a slave and its inputs and outputs are used for interaction with the physical world. Currently the Arduino UNO, Arduino MEGA2560 and Seeeduino Mega boards are supported but the sketch [3] can be easily modified for other boards.

The Arduino receives commands from the master via USB connection, performs the requested operations and responds to the master. The response may contain just a confirmation or the requested data. The communication protocol is described below in more detail.

Just like in all industrial slave devices, a watchdog timer can be enabled in the Arduino. The watchdog timer is a safety mechanism for the cases when the communication with the master is lost and every student of automatic control should be experienced in using this mechanism.

2.2 Raspberry Pi

The Raspberry Pi is a fully featured credit-card sized computer which is capable of running applications just like a standard desktop PC. The device is based on Broadcom BCM2835 SoC, which contains a 700 MHz ARM1176JZFS CPU with hardware floating point unit. Onboard memory is 512 MB. Although initially developed to increase interest in programming and software engineering, it soon became accepted as a universal programmable control unit for many machines and M2M applications. Several operating systems have been ported to the Raspberry Pi, an optimised version of Debian Linux known as Raspbian is probably the most popular one. The 35$ model B thus offers everything what is required for a modern programmable controller - enough computational power and memory, Internet connectivity and permanent data storage on a SD card.

In our case the Raspberry Pi acts as the main control unit. It is a master for the Arduino. The Arduino appears as a standard serial port upon connecting it to the Pi with the USB cable so the commands are sent as soon as they are written to the appropriate serial device.

Although several expansion boards for Raspberry Pi like Gertboard [The Raspberry Pi Foundation, 2012a], PiFace [University of Manchester, 2012] or Quick2Wire Interface Board [Quick2Wire, 2012] are available, the master-slave combination with the Arduino mimics the standard structure of large industrial control systems with remote I/O units. Plus it has additional advantages which will be discussed later.

2.3 REX Control System

The REX Control System is an open and scalable system, which is suitable for embedded control. REX can be easily ported to different platforms with C and C++ language compilers, from dedicated control cards and simple real-time executives to process stations equipped with standard operating systems. Recently it has been ported to Raspbian.

The REX Control System is formed by a family of products for the design, development and deployment of industrial control systems. By installing RexCore, the runtime of the REX Control System, the Raspberry Pi is turned into a modern programmable controller. RexCore takes care of timing and execution of individual control tasks, which can have individual priorities and execution periods.

The individual control tasks can be created in RexDraw graphical development environment or in Simulink. In both cases the control algorithms are created by interconnecting function blocks from various libraries of the RexLib industrial blockset (analog signal processing, logic control, regulation, etc.). A user-programmable block called REXLANG is also available and exactly this block has been used to implement the master part of the REXduino communication protocol. But all these details can be kept hidden from the user, who simply works with a function block and sets its parameters (see Fig. 2), which define the behavior of individual IO pins of the Arduino board. See section 4 for more details.

The REX Control System is closely connected to the PIDlab.com and Contlab.eu portals and the REXduino platform is the least expensive and the most straightforward way to test all the algorithms presented by these portals in real world.

[3] The term used in Arduino environment for the set of files forming the project to be compiled.

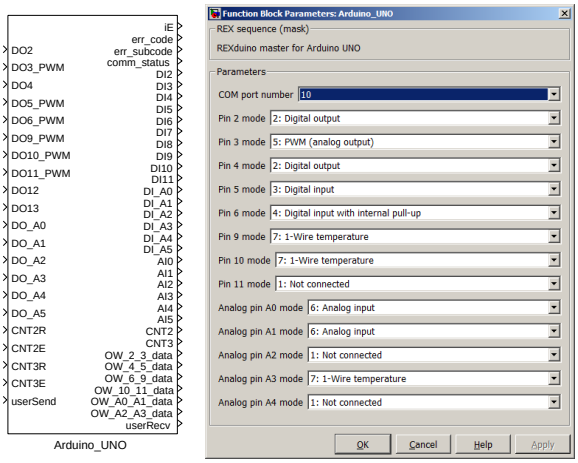

Fig. 2. Function block for Arduino UNO and its parameters

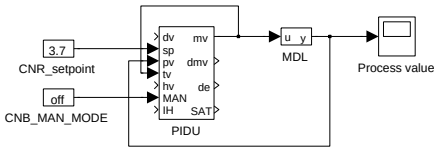

Fig. 3. Simulated PID control loop using the RexLib function blocks

3. WORKFLOW

The use of the REXduino educational platform is very easy, especially for users who are familiar with Simulink. But neither knowledge nor license of Simulink is required to get started with the development of real-time control systems. Only the simulation phase is affected if Simulink license is not available. The typical steps during the development of control algorithm are described in the following sections.

3.1 Pure simulation

The first step when creating control systems involve the analysis of the controlled system followed by the design of the control algorithm. The control algorithm is constructed from function blocks of the RexLib industrial blockset. No programming skills are required, the user just drags and drops blocks and connecting lines. A simple PID control loop based on industry-proven algorithms can be created within a few mouse clicks (see Fig. 3). The controlled system is present in the form of a mathematical model. The behavior and internal structure of the function blocks is described in more detail in REX Controls [2013a].

3.2 The first physical interaction

As soon as the simulation results comply with all requirements, we can switch from simulation to physical world. For this purpose only the Arduino board is used, it is connected directly to desktop PC or notebook via the USB cable. The model in the scheme is replaced by the REXduino master block and a SLEEP block must be introduced to run the control algorithm in "real-time" (see Fig. 4). The discrete numeric solver with fixed step size set

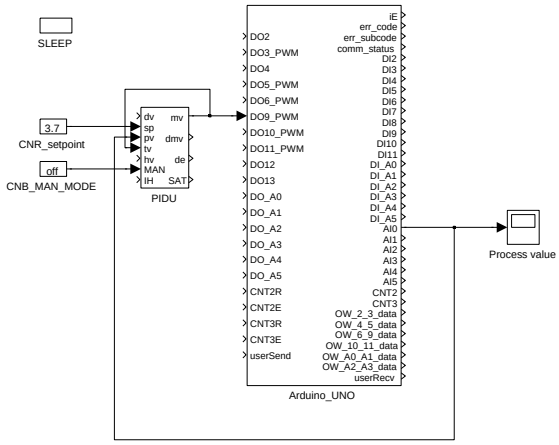

Fig. 4. PID control loop using the REXduino master block

to the same value as in the SLEEP block must be used. The step size defines the sampling period of the control algorithm.

The user can then execute the Simulink scheme just like in the case of pure simulation, only the PID control algorithm is now fed by real-world data acquired from the analog input of the Arduino board and the manipulated variable (output of the controller) is sent to PWM output.

This simple and straightforward step from simulation to physical world is especially useful for control education purposes as the students remain in the Simulink environment which is familiar to them. Moreover, the tuning of parameters or updating the control algorithm is very quick, there is neither need to compile the project nor transfer any files. Sampling frequencies of up to 50 Hz can be used, which is sufficient for many educational plants. The main factor limiting the sampling frequency is the non-real-time behavior of Simulink running on a standard PC.

This step would not be possible with Raspberry Pi-specific interface board, which is a reason why the Arduino board is so valuable for the REXduino educational platform.

3.3 Real-time control

The last step of creating a real-time control system is deployment of the algorithm on the target platform. This step is very easy with the REXduino platform as well, the only thing the user has to do is to configure the real-time executive of the REX Control System. The control algorithm remains the same as in the previous step.

Configuration of the executive consists of defining the target platform (Linux in the case of Raspbian) and the execution period of the control algorithm. The control algorithm from the previous step is referred to by the TASK block connected to the Level0 output (see Fig. 7). There can be more control tasks in general, and each of them can have its own execution period and priority. In Raspbian the REXduino platform allows sampling frequencies of up to 200 Hz, depending on the complexity of the control algorithm and number of input/output signals.

The project is then compiled by the RexComp compiler and sent to the Raspberry Pi, where it is executed by RexCore, the runtime of the REX Control System.

Natural part of the REX Control System is the ability to monitor the control algorithm in real-time and change the parameters of individual function blocks. The on-line monitoring via TCP/IP connection is available in the RexView diagnostic program or directly in the RexDraw development tool. The user is thus allowed to tune the control algorithm in the loop, without the necessity to stop it and compile it once again.

The REX Control System also offers advanced tools for data storage and further processing. The measured data and variables of the control algorithm are stored in the memory of Raspberry Pi from where it can be retrieved using the RexTrend utility. This utility reads the data and exports them in various formats including M-file for generating Matlab figures.

4. THE REXDUINO PROTOCOL

The REXduino protocol is a set of very simple commands, which define the operating modes of individual Arduino pins, set the outputs and read the inputs. Each command is terminated by a semicolon.

In the initialization phase the REXduino master starts communication and sends commands to change pin modes to the desired ones. Once the REXduino slave reports that all pins have been switched to the requested modes, the master starts polling data and updating the outputs.

4.1 Pin modes

There are several modes the pins of the Arduino can be configured for:

I - Digital input
This mode is used for reading on/off values.

J - Digital input with pullup
This mode is used for reading of e.g. buttons or switches which connect the pin to ground only when pressed. If not connected, the pin is pulled to HIGH state.

O - Digital output, default LOW
This mode is used for generating on/off signals and can be used for powering of low-consumption circuitry. The pin is set to LOW upon initialization.

Q - Digital output, default HIGH
In this case the output pin is set to HIGH by default.

A - Analog input
This mode is used for reading the 10-bit analog inputs of the Arduino (pins A0 to An).

P - PWM output
In this mode the pin generates high-frequency pulses whose duty cycle can be defined with 8-bit precision. Available only on pins marked by $\sim$.

C - Counter
The counter mode is available on pins 2 and 3. It employs interrupts and counts rising edges occurring on the pin. Pins 4 and 5 in B mode define the counting direction for pins 2 and 3 respectively.

E - Encoder signal A
This mode is used for evaluating encoder signals. It is available on pins 2 and 3. Both rising and falling edges of signal A trigger an interrupt routine. Signals on pins 4 and 5 in B mode are used to determine the direction of rotation.

B - Encoder signal B or counter DIR
This mode is available on pins 4 and 5 and must be used for proper function of counter/encoder mode on pins 2 and 3 respectively.

W - 1-Wire temperature
In this mode the pin serves as data input/output on 1-Wire bus. The temperature sensors DS18B20, DS18S20 and DS1822 are supported.

4.2 REXduino commands

Table 1 lists some of the commands and responses of the REXduino protocol. Commands for reading or setting more inputs/outputs by one command are not listed for clarity. See REX Controls [2013b] for full description.

Initialize communication							
Command:	C	0	;				
Response:	C	'0'	;				
Set pin mode							
Command:	M	nPin	mode	;			
Response:	M	nPin	mode	;			
Read analog input							
Command:	A	nPin	;				
Response:	A	nPin	BH	BL	;		
Set PWM output (analog output)							
Command:	P	nPin	value	;			
Response:	P	nPin	value	;			
Read digital input							
Command:	I	nPin	;				
Response:	I	nPin	state	;			
Set digital output							
Command:	O	nPin	state	;			
Response:	O	nPin	state	;			
Read 1-Wire temperature							
Command:	T	nPin	;				
Response:	T	nPin	status	;			
Response:	T	nPin	status	nS	B1	B2	;
Read counter or encoder value							
Command:	N	nPin	status	;			
Response:	N	nPin	status	B1	B2	B3	B4 ;
User-defined command							
Command:	U	B1	B2	B3	B4	;	
Response:	U	B1	B2	B3	B4	;	

Table 1. Commands of the REXduino protocol

The User-defined command is prepared for easy customization of the platform for individual cases (custom functions, communication with I2C or SPI devices, etc.). Both REXduino master and slave implementations are open-source thus it is quite easy to add arbitrary commands and functionality.

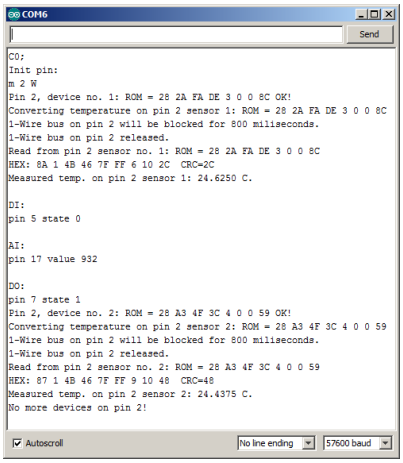

Fig. 5. Communication with the REXduino slave from the serial line monitor

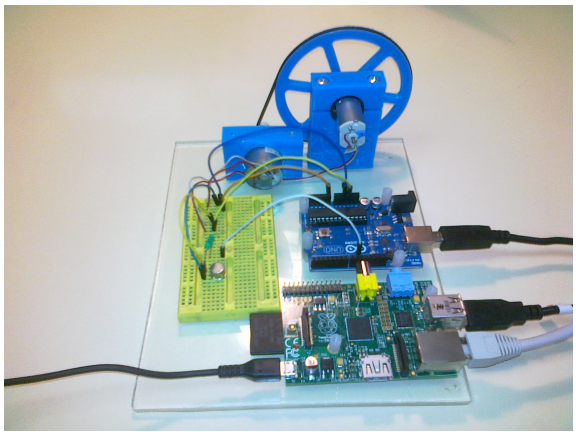

Fig. 6. Motor-generator model controlled by the REXduino platform

The majority of commands exist in two versions. The first one is used by computers and has been already listed in Table 1. The other version of the command is lower-case and is human-readable. It is intended for use in the serial line monitor, which is useful for verification of hardware connections to individual sensors as shown in Fig. 5.

5. EXAMPLES

5.1 Angular velocity control

The first system which can be controlled by the REXduino platform is a motor-generator example. One DC motor is controlled by PWM output of the Arduino which is connected to the base of an NPN transistor switching the current through the motor on and off. The other DC motor is driven by the first one via an elastic belt and generates current as it revolves (see Fig. 6).

Therefore the angular velocity of the generating motor can be measured by the analog input of the Arduino. The signal is very noisy so it is necessary to suppress the high frequencies by a Kalman-filter based filter KDER first (see Fig. 7 and then it is possible to close the angular velocity PID control loop to suppress the repetitive disturbance caused by the improperly aligned axis of the drive wheel. The effect of PID control loop is evident from Fig. 8.

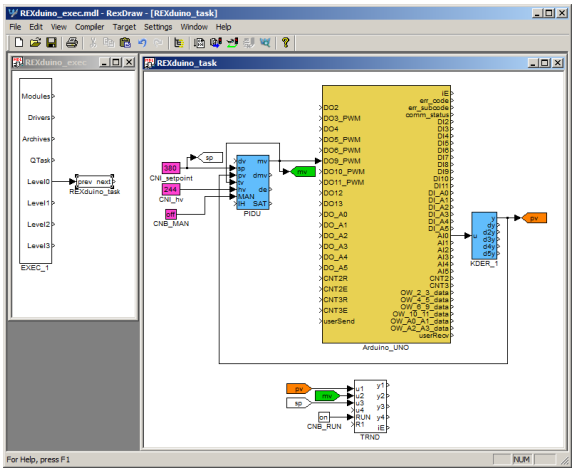

Fig. 7. Configuration of the real-time control algorithm in the REX Control System

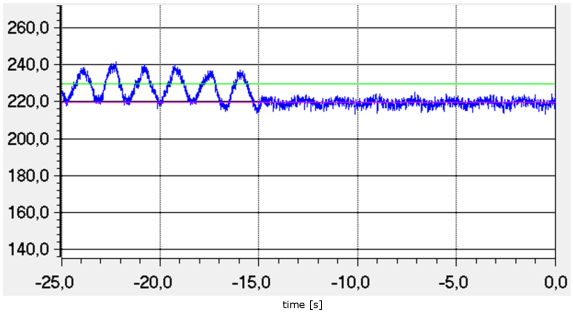

Fig. 8. Engaging the PID controller eliminates the misalignment of the drive wheel

5.2 Laboratory helicopter control

Another application for the REXduino platform was the retrofitting of an almost vintage CE150 laboratory helicopter model, which was originally controlled by a PC equipped with an ISA plug-in card. The current version of the model is controlled by a PCI card [Humusoft, 2013].

The helicopter is a 2x2 MIMO system, it has two propellers whose speed can be controlled by PWM signals. The elevation and horizontal rotation is measured by two rotary encoders. The helicopter has also a movable mass inside its body to allow dislocation of the centre of mass. The location of the mass can be changed by a small motor which is controlled by two digital outputs.

All these signals are available on the REXduino platform therefore the conversion to the new control system was straightforward and the model is fully functional after many years in the cabinet.

Although the main propeller influences mainly the elevation, the azimuth is affected as well. The tail propeller also influences both outputs having the main impact on the azimuth. In other words, there are significant crosscouplings in the helicopter model.

A cascade of P and PID controller on each propeller with a feedforward action derived from the other propeller allows stabilization of the helicopter in the given elevation while changing the azimuth, which is shown in Fig. 9.

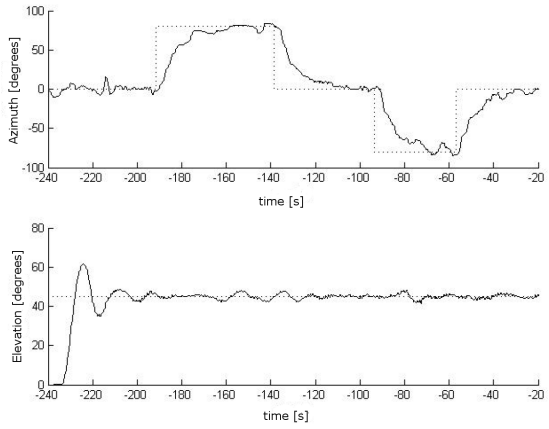

Fig. 9. Stabilization of the helicopter model and azimuth control

5.3 Electrical boiler control

The last example of use includes a set of DS18B20 1-Wire temperature sensors connected to the Arduino, which measure temperature in 5 rooms of a building as well as the outside temperature and the heating and returning water to the electrical boiler (Protherm Ray 12K). The PWM output of the Arduino is used for defining the desired temperature of the heating water and one digital output is connected to a relay which replaces the original thermostat.

The REXduino platform allowed to upgrade the heating system from simple thermostat on/off control to advanced equitherm control at negligible cost. Moreover, it allows the owner to monitor the building and the boiler remotely over the Internet.

6. OTHER USES OF THE REXDUINO PLATFORM

The REXduino platform is not limited to feedback control tasks, it can be used on the model side in hardware-in-the-loop simulations. The function block library of the REX Control System contains also function blocks for simulation of continuous-time systems, therefore it is possible to create complex models of the real plants. Again it is possible to change the parameters of the model in real-time, inject disturbances, etc., so the control system can be tested thoroughly.

Another field of application might be the control of batch experiments, where some sequence of actions has to be repeated (e.g. switching the lamps on and off at given intervals, opening a valve when target temperature is reached, etc.). Plus it can serve as a datalogger during the experiments as all the measured data can be stored in the RAM or on the SD card of the Raspberry Pi.

Beyond the scope of the REXduino educational platform, it is also possible to install additional drivers of the REX Control System to the Raspberry Pi to enhance its features. The Raspberry Pi can then act as Modbus TCP Master or Slave and exchange data with true industrial I/O devices.

7. HOLY GRAIL IN CONTROL EDUCATION?

The educational platform consisting of an Arduino board, Raspberry Pi computer and the REX Control System presented in this paper opens an extremely inexpensive and straightforward path from simulation to real-time physical computing and control. The platform requires negligible initial knowledge in programming, electronics, control theory or industrial communication standards therefore the student's first on/off control loop can be closed within minutes, which encourages the student and raises the interest. The platform is very open so it offers enough opportunities for in-depth study in any of the above mentioned fields once the student decides to do so. Both students and academicians benefit from very simple interface, steep learning curve and compatibility with Simulink and the PID-lab.com and Contlab.eu portals. The price/performance ratio is also excellent therefore the authors believe that the platform might be considered (with a little bit of exaggeration) the Holy Grail in control education.

REFERENCES

Arduino. Arduino open-source prototyping platform. http://www.arduino.cc, 2012.

Pavel Balda, Miloš Schlegel, and Milan Štětina. Advanced control algorithms + Simulink compatibility + Real–time OS = REX. In *Preprints of the 16th IFAC World Congress*, Prague, Czech Republic, 2005.

Humusoft. CE150 helicopter model. http://www.humusoft.cz/produkty/models/ce150/, 2013.

Quick2Wire. Quick2Wire Interface Board. http://www.quick2wire.com, 2012.

REX Controls. *REX system function blocks – reference manual*, 2.07 edition, 2013a.

REX Controls. *REXduino protocol – user guide*, 2.07 edition, 2013b.

The Raspberry Pi Foundation. Gertboard. http://www.raspberrypi.org/archives/tag/gertboard, 2012a.

The Raspberry Pi Foundation. Raspberry Pi. http://www.raspberrypi.org, 2012b.

University of Manchester. Pi-Face Digital Interface. http://pi.cs.man.ac.uk/interface.htm, 2012.