Educational Robotic Platform based on Arduino

Richard Balogh
Faculty of Electrical Engineering and Information Technology
Slovak University of Technology in Bratislava
richard.balogh@stuba.sk

Abstract — **The design of a new controller board for a mobile robot based on the Parallax Boe-Bot chassis is described. Disadvantages of the original Basic Stamp processor disappeared, more complicated tasks can be solved. As the board is compatible with the Arduino platform, also the open source development environment can be used. The requirements, design process and technical parameters are described. Also some illustration examples are shown.**

Keywords – controller, mobile robot, Arduino, robotic platform

I. Introduction

In our university we have been using for many years the commercially available mobile robots Boe-Bot[1] by Parallax, Inc. for education [1]. They were used in some laboratory exercises for students of the Mobile robotics lectures, some additional lectures for students of Embedded systems, or Automotive control systems. We were using this platform also for summer courses, student projects and for public presentations.

Our main problem with the Boe-Bot robot was with its controller unit. Although the Basic Stamp II with its programming capabilities is very reliable and useful for start up, our advanced students at the University were critical to use BASIC as a programming language for robots. They lack function definitions, program hierarchy, interrupts, parallel tasks, and direct access to peripherals like timers, counters etc. Our experiences with 8-bit RISC AVR processors by Atmel and growing popularity of the Arduino platform lead us to the design of a completely new controller board for robots. The main goal was to achieve as much compatibility as possible. Not only dimensions (which are essential to replace the board for the original one), but also the overall concept, connectors placement etc. were sustained. Now, we can use the robot with almost all original extensions of the Boe-Bot robot.

II. Components of the system

The Boe-Bot mobile robot [2] is a commercially available robotic kit by Parallax, Inc. It consists of two geared motors mounted on an aluminium chassis, batteries and control electronics. On the motors, there are mounted two plastic wheels. The rear wheel is made of a drilled polyethylene ball. Mounting holes and slots may be used to add custom robotic equipment.

The robot is controlled by the Parallax's popular microcontroller Basic Stamp II and the Board of Education. It is a simple board containing a processor, power supply circuits, interfaces, connectors and a small experimental solderless breadboard. The Basic Stamp II processor can be programmed with the PBASIC language - simple, but powerfull clone of BASIC language with a support of many specific peripheral devices [2]. Pros and cons of this platform were evaluated in details in [3].

Arduino is an open-source electronics prototyping platform based on a flexible, easy-to-use hardware and software. It is intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments [4]. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing) [5]. The hardware reference designs (CAD files) are available under an open-source license, anyone is free to adapt them to its own needs.

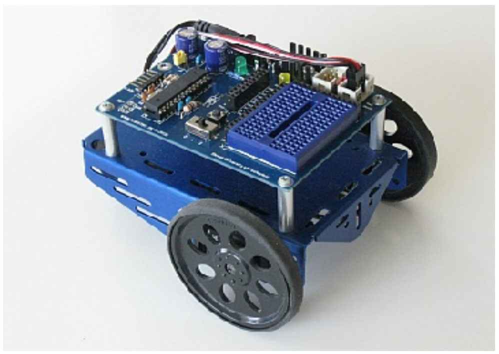


Figure 1: The Boe-Bot mobile robot with the new controller.

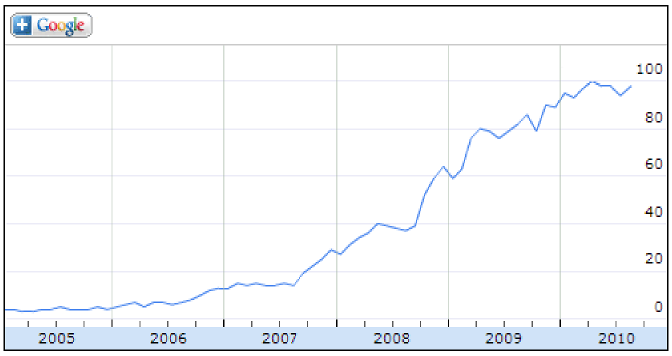


Figure 2: The number of searches for the term 'Arduino', relative to the total number of searches done on Google (source: Google Insights for Search).

1 http://www.parallax.com/Store/Robots/AllRobots/tabid/128/ProductID/302/List/1/Default.aspx

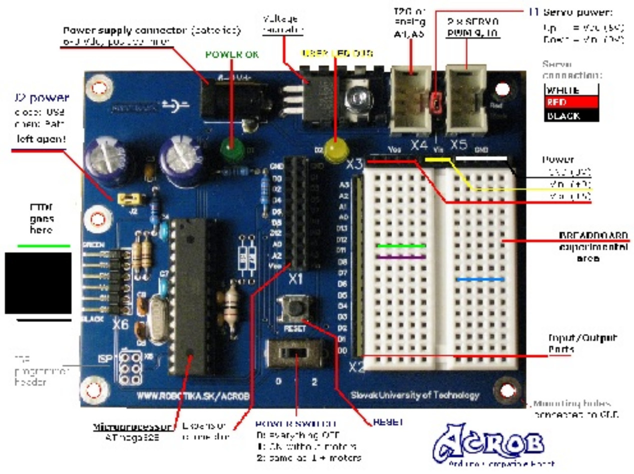


Figure 3: Description of the board.

This is also our case, we designed a completely new board retaining the compatibility with the platform. Arduino growing popularity (see Fig. 2) provides us with great supporting community with constant development of the software, many examples, tutorials and projects available.

III. Hardware

The new board is as much compatible as possible with the original Board of Education by Parallax Inc. [6]. The main differences are two: a different processor and a TTL serial interface without the converters.

As a power supply we can use the battery box (with four primary or rechargeable 1,5V AA size batteries) or a wall adapter. The power switch has three positions. Except the standard on/off positions there is a special "development" position when motors are disconnected so the robot is not moving on the desktop during the debugging.

An on-board voltage stabilisator provides 5V for the microcontroller and its peripherals. As the main processor, Atmel Atmega328P with a pre-burned bootloader is used. It provides the user with 32kB of the program memory, 2kB of data RAM space and 1kB of the EEPROM space. The main area of the board is occupied with a solderless experimental breadboard which enables to connect different additional components. On its left side most of the I/O pins are available, on its top there is a power supply connector. The board also contains connectors for servomotors and two additional sensors with digital or analogue outputs. A dual line connector in the center of the board enables to connect standard Parallax's extension boards like the compass or the LCD modules. See also the description of the comprehensive set of connectors for peripherals contained on the board in the Tab.1.

Programming and communication capabilities were increased comparing to the original Boe-Bot robot. We decided to have only the serial communication interface with TTL levels without any other converters on the board, so different converters can be used. We can use the standard

TABLE I. Connector Description Table

ID	*Purpose*	*Type*
X1	Expansion connector compatible with the Parallax AppMods.	2x10
X2	Access to the I/O pins	1x16
X3	Power supply for breadboard (Vcc, Vin, GND)	1x13
X4	Sensors (I2C bus and/or Digital/Analog Input)	2x3
X5	Servomotors (2xPWM)	2x3
X6	Serial / Programming interface (bootload)	1x6
X7	Power supply jack	3,5mm
X8	In System Programming interface – ISP (MISO, MOSI, RESET,...)	2x3

FTDI Chips[2] USB cable or the SparkFun's FTDI Basic[3] module for programming using the internal bootloader. Also we developed RS-232 level converter module to enable operation also with a standard serial interface (see the Fig. 4).

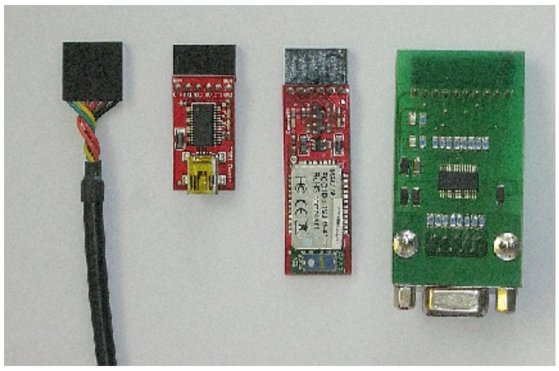


Figure 4: Programming and communication using (from left to right)
a) FTDI USB Cable, b) SparkFun FTDI Basic module c) SparkFun Blue Mate Bluetooth module d) custom made RS-232 module.

After the program loading, the interface is free for any user serial communication operations. This enables to connect e.g. SparkFun's BlueMate[4] communication module to communicate with the computer or between robots using the Bluetooth interface. On the board there is also the connector for ISP programmer, so one can use any standard Atmel ISP programmer to burn the program into the processor. Together with the AVRStudio one can even debug, step and watch programs written in Assembler or avr-gcc languages.

The board can be used also standalone with the only connection using the FTDI USB cable. In such case user can power the board from the USB interface so no additional equipment is necessary. Such configuration can be used for introduction to the embedded systems programming, explaining basics of digital and analogue inputs, outputs or built-in peripherals like timers, counters, PWM and A/D converters. The schematics and the printed circuit board were

2 http://www.ftdichip.com/Products/Cables/USBTTLSerial.htm
3 http://www.sparkfun.com/commerce/product_info.php?products_id=9115
4 http://www.sparkfun.com/commerce/product_info.php?products_id=9358

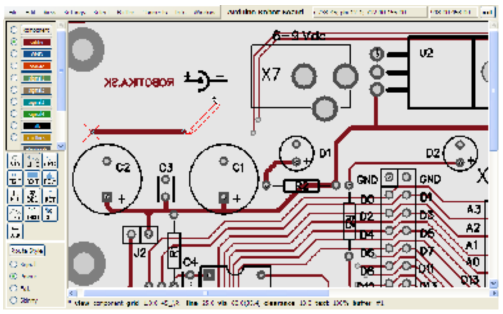


Figure 5: Design in the pcb program from the gEDA suite.

designed using the open source gEDA suite[5]. We used thru-hole components to enable students to build their own boards from the kits.

IV. Software

For programming, the standard Arduino IDE can be used. Other methods include the Assembler or avr-gcc languages integrated within the Atmel AVR Studio or using a set of command line utilities. We tested the environment on MS Windows XP operating system, but the Arduino IDE should work also on Linux and MAC OS systems. There is only one problematic point we found - during the installation process one need administrative rights to install USB drivers properly. This problem diminished in Windows 7 where drivers seemed to be already contained.

We prepared a set of basic programs to show an access to peripherals. We start with a basic digital I/O (LED and switch), then move to the analogue world – basic robot movements and analog sensor measurements. As the first analog sensor we find very useful Sharp distance sensors which are easy to connect and offer reliable results. Also their non-linear characteristics is challenging.

As the very first program we used the standard "Hello, World!" problem.

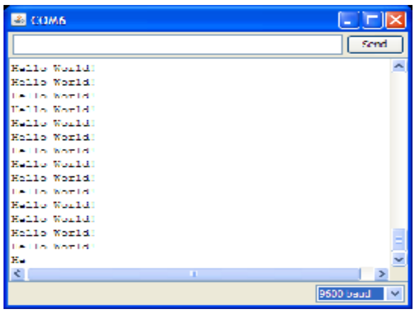
```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
 Serial.println("Hello, World!");
}
```

After the compilation and burning the program into the processor using the bootloader, a user can see the result using the internal built-in terminal window (see Fig. 6). Sometimes communication speeds didn't correspond to the real ones and characters were displayed incorrectly until it was changed.

From the listing above it is clear that programming is very straightforward and at the beginning no processor specific knowledge is required. Only important thing is to split the program into two basic parts – **`setup`** (which is performed only once) and **`loop`** (which is then performed infinitely – or better said, until it isnot switched off or reprogrammed). As the Arduino language is built over the standard avr-gcc compiler, we can

5 http://www.gpleda.org



```
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
He
```

Figure 6. Hello World program in terminal window.

we can still use all its features and combine also the standard approach e.g. direct access to all processor registers:

```
TCCR0A |= _BV(WGM01) | _BV(WGM00);
OCR0A = 127;
OCR0B = 255;
```

Of course, libraries can hide the internals from the user so no special knowledge is required. An example of a library for servos to show basic robot movements follows:

```
#include <Servo.h> // this program uses the Servo library

Servo LeftServo;  // create servo object to control both servos
Servo RightServo; // a maximum of eight servos can be created

#define FAST  50  // try to change these values during the test
#define SLOW   5

void setup()
{
 LeftServo.attach(9);  // attach servo on pin 9 to the servo object
 RightServo.attach(10); // attach servo on pin 10 to servo object
}

void loop()
{                                // FAST FORWARD
 LeftServo.write(90 + FAST);     // value 90 is in middle, i.e. stop
 RightServo.write(90 - FAST);    // mirrored position
 delay(1500);                   // go fast forward for  1,5 s

 LeftServo.write(90 - FAST);     // ROTATE (PIVOT) LEFT
 RightServo.write(90 - FAST);
 delay(1500);

 LeftServo.write(90 - FAST);      // FAST BACKWARD
 RightServo.write(90 + FAST);
 delay(1500);

 LeftServo.write(90);            // STOP both motors
 RightServo.write(90);

 for(;;)  ;                      // stop the program operation here

} /* End of Loop */
```

For a comparison – a similar program written in the original Basic Stamp II language may look like this:

```
' {$STAMP BS2}
' {$PBASIC 2.5}

counter    VAR Word
pulseLeft  VAR Word
pulseRight VAR Word
pulseCount VAR Byte


' Forward
pulseLeft = 850: pulseRight = 650: pulseCount = 64: GOSUB Navigate

' Left turn
pulseLeft = 650: pulseRight = 650: pulseCount = 24: GOSUB Navigate

' Backward
pulseLeft = 650: pulseRight = 850: pulseCount = 64: GOSUB Navigate

END
```

```
Navigate:

 FOR counter = 1 TO pulseCount
  PULSOUT 13, pulseLeft
  PULSOUT 12, pulseRight
  PAUSE 20
 NEXT
 PAUSE 200

 RETURN
```

V. Support

We created the supporting page at our robotics server[6] with all necessary documentation. Also we have started with creation of a comprehensive manual with example programs and connection diagrams. One of the big advantages of the open source system is a large community of users adding their experiences to the whole system. As the board is Arduino compatible, we can immediately start to use an existing repository of examples, documentation etc. If You want, for instance to connect an ultrasonic detector SRF-08 to a robot, you find very soon not only few examples but also a connection diagram[7] and even the whole library[8] for this sensor. Just type keywords 'SRF08' and 'Arduino' to your favourite search engine.

VI. Evaluation

A new robot, called Acrob (**Ar**duino **C**ompatible **Rob**ot) was tested and evaluated at some different events. We prepared the robotic introductory lecture for students of the Automotive branch of study. The main goal was to give them an idea of mobile robots and its programming. During two lectures students were able to program basic movements and reactive behaviour of the robots. For students of the Mobile robotics course we prepared similar lecture and the platform was also used as an example of a differential driven platform. Also the infrared sensor distance detection was explained. Then we use the robots in a joined Austrian-Slovak lecture for secondary school students. During the lecture they were able to program movements, connect and evaluate line sensors and distance sensor so they finished with a simple line-following robot with an obstacle avoidance. In the time of writing this paper the robots weere intensively tested in the Summer School of Robotics and they were succesfull. The overall concept was succesfully tested also at the contest Robotchallenge Wien 2010 where the robot succesfully (though very slowly) passed the linefollowing category (see Fig. 7).

VII. conclusions

Presented robotic platform offers many capabilities. The main problem of the previously used Parallax's BoeBot platform – programming in BASIC was successfully solved and programming is now possible both in assembly and C++ languages. Moreover we can use a large repository of examples and tutorials for the Arduino, libraries and components for this platform and also large community shared resources. The concept was proven in some robotic lessons for university and secondary school students and also at the summer school and robotic contests.

6 http://www.robotika.sk/acrob

7 http://www.robot-electronics.co.uk/htm/arduino_examples.htm#SRF08%20Ultrasonic%20Ranger

8 http://www.arduino.cc/playground/Main/SonarSrf08

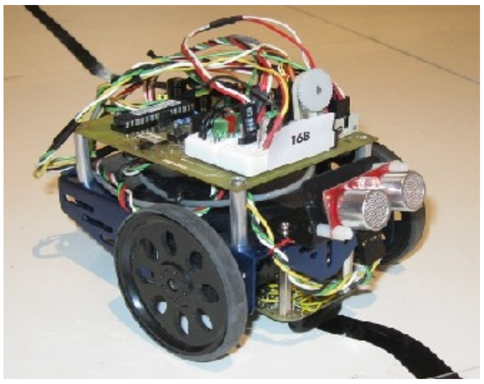


Figure 7: Prototype of the robot with linefollowing sensor and ultrasonic obstacle detector at the Robotchallenge contest.

Acknowledgment

This project is a part of the project CENTROBOT funded by the EFRE program Creating the future (CBC SK-AT 2007-13). Author also thanks to Parallax, Inc. for help and support.

References

[1] Balogh, R.: *Basic Activities with the Boe-Bot Mobile Robot.* In DidInfo 2008. 14th International Conference. FPV UMB, Banská Bystrica, Slovakia.

[2] Lindsay, A.: Robotics with the Boe-Bot. Student guide. Version 2.2., Parallax, Inc. Rocklin, California, 2004. ISBN 1-928982-03-4. Available on-line: http://www.parallax.com

[3] Hofmann, A. et al.: Robotic Education Platform. Literature Research and Evaluation. Technical Report, part WP2 of the Centrobot project. Wien, 2009. Available on-line: <http://www.centrobot.eu/en/aktivitaeten/robotic-education-platform/>

[4] Arduino Home Page. http://www.arduino.cc/

[5] Banzi, M.: Getting Started with Arduino. O'Reilly, 2009.

[6] Board Of Education. Development / Educational Platform for the BASIC Stamp and Javelin Stamp Microcontrollers. Parallax, Inc. 2007. Available on-line: <http://www.parallax.com/Portals/0/Downloads/docs/prod/boards/28150-BOE-Serial-v2.0.pdf>