AN INTEL GALILEO PLATFORM APPLICATION DEVELOPMENT USING MATLAB

Eugen LUPU[1], Radu ARSINTE[1], Horea CHIS[2]
[1]*Technical University of Cluj-Napoca,* [2] *NTT Data Cluj-Napoca*
Eugen.Lupu@com.utcluj.ro

Abstract: MATLAB is a powerful tool often employed by researchers in order to simulate or test their applications. Most often the applications don't interact with other external hardware, but MATLAB has facilities to communicate with external devices by the means of the I/O ports. The paper presents an approach for control applications which run on MATLAB and monitor a process interfaced to a Galileo platform. The resources provided by MATLAB allow the user to change and experiment very easily with different control algorithms and to benefit of a very friendly GUI for the applications. This type of application is targeted to educational purposes.

Keywords: *MATLAB, educational application, Intel Galileo platform, control algorithm, Arduino*

I. INTRODUCTION

Educational applications require the use of simple hardware and software platforms, flexible and inexpensive allowing rapid deployment and able to make changes and adjustments effortlessly.

Intel Galileo G1 ("first generation") is a board meant for the development of applications, which is based on the Intel Quark SoC (System-on-Chip) X1000 32 bits. Intel Galileo is designed as a flexible development environment used as the first stand based on Intel architecture, developed hardware and software compatibility with the Arduino platform. This enables the use of the most popular platforms for developing embedded systems and applications for them, Arduino, resulting in flexibility in design applications, access to a large amount of open-source code, community or support and enthusiastic, able to help on various issues. [3]

Available hardware platforms are easy to use, designed both beginners and hobbyists. Operations that reading input pins, activation of LED lighting for a motor or can be executed by using the Arduino programming language and the integrated development environment (IDE) with the same name.[1]

Arduino concept was originally developed as a tool for rapid prototyping, for students with limited knowledge of electronics and programming. As time Arduino community has expanded and so changes have occurred occurring variants of platforms, from the simplest 8-bit, to more complex.

Currently there are other more advanced platforms for vital applications, including Parallax Basic Stamp, Basic Tiger, Raspberry Pi or the Beagle Board. However Arduino platform has some advantages that made its popularity grow to the detriment of others. The most important advantages are the low cost of hardware models offer compatibility with most operating systems existing environment and programming language simplicity and open-source software and hardware support. [1]

Matlab programs provide opportunities intended to run on an Arduino platform. Through a program running on the microcontroller on the Arduino, and a package of support that allows the transmission of commands in Matlab serial port, it is possible to interface the two platforms for the development of various applications and for use for educational purposes.

Using Matlab environment for applications running on Arduino has some advantages over the classic use development environment dedicated. First would be the possibility of starting with Arduino programming without the need for other software installed to facilitate this. Another advantage is the ease of understanding commands that may be sent from Matlab to be interpreted by Arduino, and these suggestive names (e.g. digitalRead, analogRead or digitalWrite). Lastly, it is taken into account for calculating the advantages of improved performance. [5]

Through a dedicated support package, Matlab allows communication with an Arduino development platform via a USB cable. The support package is based on a program running on the Arduino microcontroller and receiving commands through the serial port, executes them, and if necessary, returns a result. This approach allows developing the applications without the need of any additional tools using Matlab for interactive development and debugging, development of programs that allow data acquisition and analog and digital control of: parameters (e.g. temperature), various types of motors (DC servo or stepper) or other processes. [8]

II. THE INTEL GALILEO PLATFORM

Intel Galileo platform is a real Linux OS based microcomputer rather than an Arduino compliant board. By the Ethernet interface and a mini-PCIe extension slot the connectivity of the board can be extended. This mixture of hardware and software allows to Intel Galileo to communicate with other devices in intranet or Internet

Manuscript received November 12, 2016; revised December 8, 2016

employing wired or wireless interfaces, just like a regular computer. [1]

The Intel Galileo platform is also software compatible with the Arduino Software Development Environment. In contrast to Arduino board hardware, the Galileo board has several PC industry standard I/O ports (Micro-SD slot, RS-232 serial port, USB Host port, USB Client port) and features to extend the usage and capabilities beyond the Arduino shield repository. So, for complex applications to use the Galileo platform became more convenient avoiding the use of additional extension boards. [2]

A top view of the Galileo board can be seen in the fig.1 where around the Intel Quark SoC X1000 one can count a lot of peripheral able to facilitate the board connectivity.

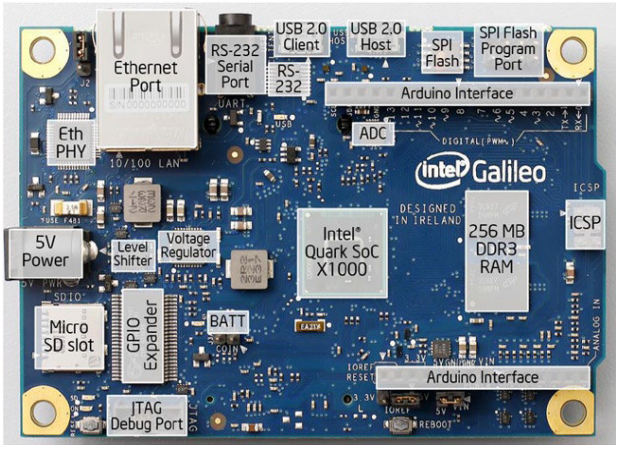

Figure 1. Top view of Intel Galileo board [1]

Some of the most important additional hardware/software features brought by Intel Galileo are resumed below and all are accessible by the standard Arduino's library, so:

- works with PCI Express Mini Cards (Wi-Fi, Bluetooth, GSM cards for connectivity
- USB Host Port (USB On-The-Go port)
- MicroSD Support
- I2C, SPI Support
- Serial Connectivity
- Shield Compatibility (5V and 3.3V Arduino shields)
- Arduino familiar IDE
- Ethernet Library Compatibility
- Real Time Clock
- Linux on Board (one can boot Galileo from a SD card image that Intel provides) [2].

III. THE APPLICATION IMPLEMENTATION

The block diagram of the application developed on the Intel Galileo platform and controlled by a MATLAB program can be seen in the next figure. The Galileo board is directly connected to the process (in this case a temperature regulation using different control algorithms) by the analog and digital I/O of the board.

Mathworks provide for the users the MATLAB Support Package for Arduino hardware, so we can use MATLAB to communicate with an Arduino board over the USB. The package allows performing tasks such as:

- acquire analog and digital data from different sensors from the Arduino board
- to access peripheral devices and sensors connected over I2C or SPI
- to control devices with digital and PWM outputs
- to drive DC, servo, and stepper motors etc. [8][5].

Using Arduino or other compliant board and the support package in MATLAB is possible to run programs via the Arduino microcontroller.

Matlab commands are transmitted to the development board and these commands are interpreted by a program "server" running on the Arduino microcontroller. In order to use this support package, it first needs to charge on Arduino the program "server", which will run continuously and will translates the orders received and will execute and, if necessary, will return results. The next step is to connect the platform to a USB port on the host computer, and if everything works correctly with the command:

```
a = Arduino ( 'port')
```

, where 'port' is a COM virtual port seen by the computer will perform the connection between Matlab and Arduino. Other commands that can be transmitted by Matlab to Arduino are presented on the website of Arduino Matlab support package, presented in [6].

Figure 2. Galileo platform connection via a COM port

The application running on the compliant Arduino board is monitored by the MATLAB application which sends different commands and parameters to the platform in order to select the desired control algorithm.

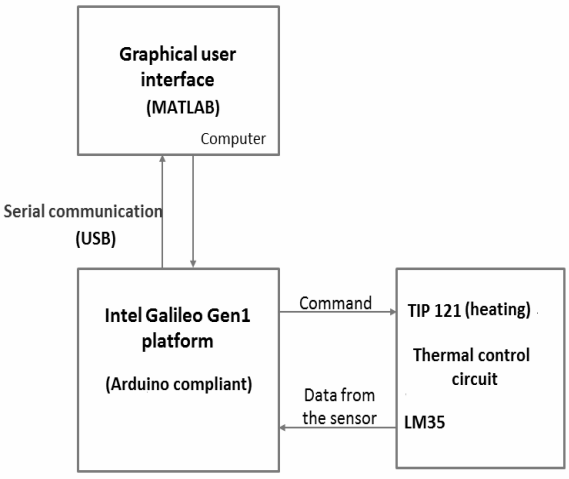

Figure 3. The block diagram of the application

In order to have reliable data exchange between the two applications a hand-shake protocol was employed fig.4. The communication protocol referred to hand-shake routine is executed each time data is transmitted between the two platforms. This routine makes it possible to synchronize the timing of the reception and transmission with reverse, thus avoid possible errors in communication.

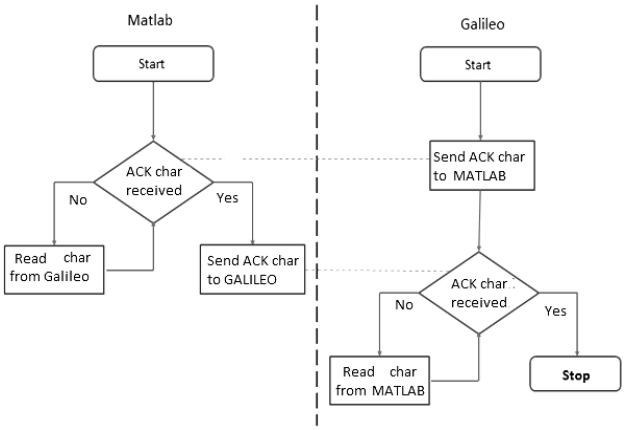

Figure 4. Matlab-Galileo hand-shake protocol

The Matlab code which performs the hand-shake process is presented below.

```
% Send the start communication character
  fprintf(serial_Connection, '%c', startChar);
% Start the acknowledge routine
  while (charCompare ~= ACK_CHAR)
    charCompare =
  fread(serial_Connection,1,'uchar');
    pause(0.02);
  end
% Send back the acknowledge character
  fprintf(serial_Connection, '%c', ACK_CHAR);
```

In order to implement the application of temperature control some additional hardware is needed to be connected to the Galileo board. The circuit shown below is connected to the board. The enclosure to be thermostatic is composed by a hard drive box. Therein the circuit is placed: a LM35 sensor for the thermal control circuit, two power resistors for heating and a TIP 121 Darlington transistor as a switch for the power to the heating elements. A cooler is employed in order to uniform the temperature inside the box. [10][13]

IV. EXPERIMENTS AND RESULTS

In order to control the thermal process a final version of the GUI for the Matlab application was implemented.
The interface is developed in accordance with algorithms for automatic temperature control (in this case the algorithm on-off and proportional algorithm). This version contains the interface temperature field, parameter setting for control algorithms and plot graph of temperature over time.

The common parameter for the control algorithms that can be set through the interface is the "prescribed temperature [°C]." For the "on-off" algorithm can be set the "hysteresis [°C]" while for the proportional algorithm the GUI provides the ability to set the "proportionality factor". The "prescribed temperature" (Tp) is not a specific parameter for one control algorithm, this is necessary for all automatic control systems. In this case the "prescribed temperature" is a general parameter and represents the threshold at which the temperature of the enclosure is adjusted by the two algorithms [4].

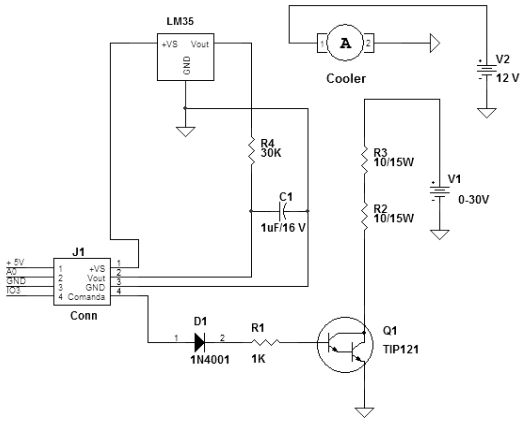

Figure 5. The thermal control circuit

The specific parameter for the "on-off" control algorithm is the hysteresis. This parameter indicates the deviation in °C from the reference temperature inside the enclosure. When Tm < Tp – hysteresis the system start to heat (ON) and when the Tm < Tp + hysteresis the system stop to heat (OFF) [4]. In automation systems, hysteresis is introduced into the development phase, as an assumed error and it aims to reduce the number of stops and starts of the adjuster. The proportional algorithm offers the choice of the parameter called factor of proportionality. This parameter of the proportional algorithm is the amount by which the error is weighted by the size in the command value. So this parameter will weigh the duty factor of the PWM command during the heating process [11].

Sending the algorithm parameters from the Matlab application to the Galileo platform is detailed below.

```
% --- Send the parameter over serial to Galileo
function Send_Parameter(serial_obj,command, value)
% serial_obj - serial connection
% command - the sent command (e.g 'S' - set command)
% value - sent parameter
% Verify the serial
      CheckSerial();
% Serial send read request to Arduino
      fprintf(serial_obj,'%c', command);
      pause(0.005);
      fprintf(serial_obj,'%s', value);
end
```

The main commands accepted and interpreted by the two environments and their values are presented in the table 1. For the previous presented system some experiments were

made.

So for the "on-off" algorithm we fixed the prescribed temperature to different values and also for the hysteresis. In fig. 7 one can see the system running trying to keep the value of the enclosure temperature to 35°C for a hysteresis of 0.5°C. If we analyze the temperature evolution some overshoots can be observed due to the thermal inertia of the system and the limitations of this algorithm.

Table1. The commands used for data transfer between Matlab application and Galileo board

Nr.	Command	Value
1	Start communication	's'
2	Sync character (ACK)	'a'
3	Set prescribed temperature	'P'
4	Set hysteresis	'S'
5	Read temperature	'R'
6	Set proportionality factor	'F'
7	Stop heating	'Q'

In fig.8 may be followed the temperature evolution for the proportional algorithm having a prescribed temperature set to 35°C and value of 20 for the proportionality factor. The temperature overshoot in this case are smaller.

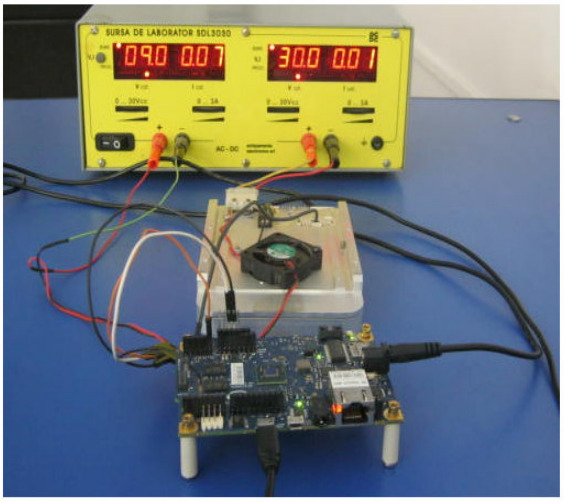

Figure 6. View on the experimental model

V. CONCLUSIONS

The paper presents an approach to develop educational control application using compliant Arduino platform (Intel Galileo) driven by a Matlab application in order to benefit the capabilities of this powerful tool: friendly GUI, different toolboxes and support packages. Also the Arduino community provides a huge database of application and software support.

A minimal hardware was added to the Intel Galileo board for performing the experiments, fig. 6. Two algorithms were been implemented in order to control the temperature in the enclosure. The experiments allow seeing and understand the possibilities and limitations of these algorithms. The additional hardware allows selecting the prescribed temperature below 65°C. Also the role of the cooler may be observed during the experiments.

Employing the same hardware resources other control algorithms may be implemented and tested.

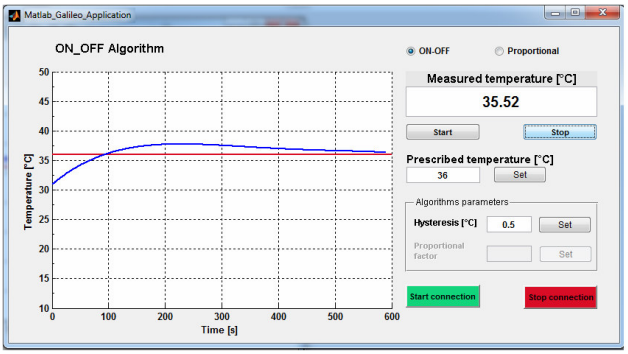

Figure 7. Temperature evolution for the "on-off" algorithm

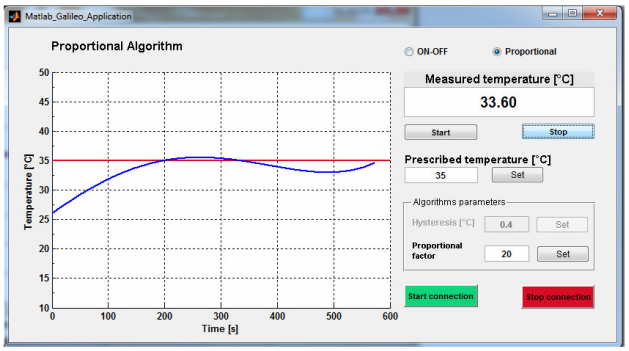

Figure 8. Temperature evolution for the proportional algorithm

Developing this type of application the users discover the manner to realize Matlab application which interact with hardware resources.

Using other additional hardware and developing the appropriate software several types of applications can be implemented (e.g. motor control applications).

REFERENCES

[1] M. C. Ramon, Intel® Galileo and Intel® Galileo G2 API Features and Arduino Projects for Linux Programmers, Apress Open, 2014
[2] "Galileo data sheet," Intel Corporation, 2014, Santa Clara, SUA
[3] "Intel® Quark SoC X1000 data sheet," Intel Corporation, Santa Clara, SUA, 2013
[4] N. Palaghită, D. Petreus, C. Fărcas *Electronică de comandă si reglaj*, Ed. Mediamira, Cluj-Napoca, 2006.
[5] http://www.mathworks.com/products/matlab/
[6] http://www.mathworks.com/help/instrument/serial-port-overview.html
[7] https://www.arduino.cc/en/Main/ArduinoBoardUno
[8] http://www.mathworks.com/hardware-support/arduino-matlab.html
[9] "AD7298 ADC data sheet," Analog Device, SUA.
[10] E. Lupu, R. Bălan, P.G. Pop "Low-cost application for algorithm control study" *Proceedings of IEEE-TTTC 2002*, pp.332-335
[11] https://www.arduino.cc/en/Tutorial/PWM
[12] https://www.arduino.cc/en/Reference/AnalogRead
[13] H. Chis, "Thermostat with MSP-EXP430G2" diploma thesis, Faculty of Electronics, Telecommunications and Information Technology, Cluj-Napoca, 2012.