Real Time Interfacing & Control techniques using an open source

Jheel Shah, Prof Vinod P. Patel

***Abstract*—This paper shows one way of controlling any system through development board with the help of open source software Scilab. This paper presents a low-cost, reusable, reconfigurable platform that enables integrated design and implementation of embedded control systems. To minimize the cost, free and open source software packages such as Linux and Scilab are used.**

***Keywords*—Scilab, Linux operating system, Real time control, Development Board**

I. INTRODUCTION

The cost of proprietary software, such as Matlab, is often expensive. One solution for this is to use Scilab which is free and open source software (FOSS) having powerful benefits and simplicity in hardware interfacing, ease in implementation of various soft controllers, etc. [5]

Scilab is a freely distributed and open source scientific software package providing a powerful open computing environment for engineering and scientific applications. It was developed at INRIA as part of the Meta2 project and includes hundreds of general purpose and specialized functions for numerical computation, organized in libraries called toolboxes that cover such areas as simulation, optimization, systems and control, and signal processing. [1] Scilab includes hundreds of mathematical functions. It has a high level programming language allowing access to advanced data structures, 2-D and 3-D graphical functions.

Linux is one of popular version of UNIX operating system. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX. [17]

Embedded systems are playing an increasingly important role in control engineering. Despite their popularity, embedded systems are generally subject to resource constraints and it is therefore difficult to build complex control systems on embedded platforms.

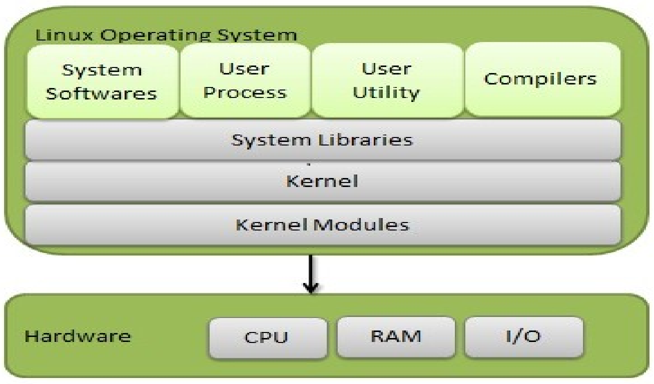

Figure 1. Block diagram of Linux Operating Sys*tem* [17]

Traditionally, the design and implementation of control systems are often separated, which causes the development of embedded control systems to be highly time consuming and costly. As this trend continues, the old way of developing embedded control software is becoming less and less efficient. Thus a low-cost, reusable, reconfigurable platform is developed for designing and implementing embedded control systems based on Scilab and Linux, which are freely available along with source code. The platform can be built on different development boards. [4]

The organization of this paper is as follows: Section II describes the different methodologies for interfacing. Section III describes interfacing and controlling using different development boards. Section IV concludes the paper.

II. DIFFERENT METHODOLOGIES

A. Java Interface for Scilab[2]

Scilab can be interfaced with Java based on the jLab environment. Here the interface is packaged as a Jar File by the Scilab provider. So we can say, the service afforded by interface is a set of method.

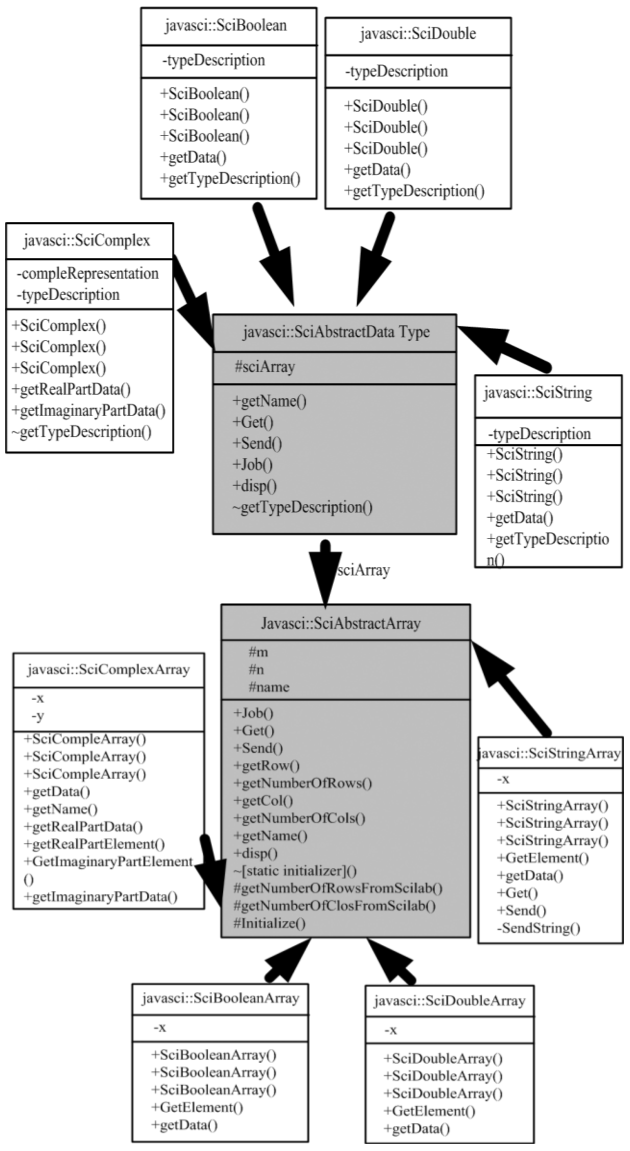

Figure 2. The architecture of the javasci.jar [2]

The Jar File consists of fourteen classes as follows:[7]
1. BadDataArgumentException
2. ClassPath
3. LibraryPath
4. SciAbstractArray
5. SciAbstractDataType
6. SciBoolean
7. SciBooleanArray
8. SciComplex
9. SciComplexArray
10. SciDouble
11. SciDoubleArray
12. Scilab
13. SciString
14. SciStringArray

B. Using Scilab for building of Virtual Lab[3]

This is one way to use Scilab software in the Internet environment. It is based on the developed client-server architecture. Since Scilab does not offer any Internet interface it is necessary to find an alternative way to exploit it. Communication can be through "shared file", "Java Interface", "TCP sockets", "pipes between processes". Here communication via TCP sockets is chosen.

The Scilab does not contain a tool for socket communication. The solution is offered e.g. by the library Socket Toolbox for scilab created by T. Reveyrand. [6] The toolbox allows to connect on a listening TCP socket and to send and receive data via this socket. The only disadvantage of this toolbox is the fact that it doesn't allow creating a listening socket. It only allows to connect to a socket that already has the status of "listening". However, this problem can be easily solved since the most of server side scripting languages, like e.g. PHP language, supports packages, toolboxes or extensions allowing a developer to gain full control (including creation and destruction) of TCP connections. The data transfer between server and client application can be accomplished using sockets that enable direct approach to network protocols of lower level. Actually, a socket creates a single unique connection between two applications.

1) Server Side

In general, a server represents the program application that delivers a service to clients. It processes all requests and commands and if it is required it ensures communication with other external applications. Since we are going to communicate with SciLab, in our case the external application is presented by SciLab environment. Actually, the server application has 4 main tasks:

- It has to create, maintain and close connection with SciLab.
- It has to reply to all requests from clients.
- It sends data to SciLab.
- It receives data from Scilab.

The core of the server side was developed in php scripting language. The developed script (server.php) has to create the socket to establish a unique connection with SciLab. Then, the script executes SciLab program environment and automatically also the script inside of SciLab (service.sce) to open the socket connection using the socket that was created by the php script. When the connection is accepted also by the server.php script, everything is prepared for the communication. After the script server.php receives preprocessed data from the client application it sends the requested instructions as astring to SciLab. The SciLab executes the received instructions and sends back the computed result. It is also in a string form that is sent for visualization to the client part of the whole application

2) . Client Side

The client application takes care about the interaction with a user. It sends commands and parameters from the user to the server and results from the server to the user.

After the user enters and submits input parameters to the form on the webpage, the request for their processing is accomplished. The client side of the application is again developed using php language. The script (client.php) transforms the input from the user into SciLab instructions and sends it via session variable to the script server.php. After request processing the client.php script receives results from the server script in a string form. This data are used for visualization of results for the user. The dynamically generated web page is created using JavaScript and CSS in combination with Document Object Model approach. For the graphical presentation we used more and more popular SVG format. It Allows to process current data from server dynamically and it gives us possibility to enhance velocity, flexibility and interactivity of the generated graphics, e.g. signalize status of Data, or update current data in time.

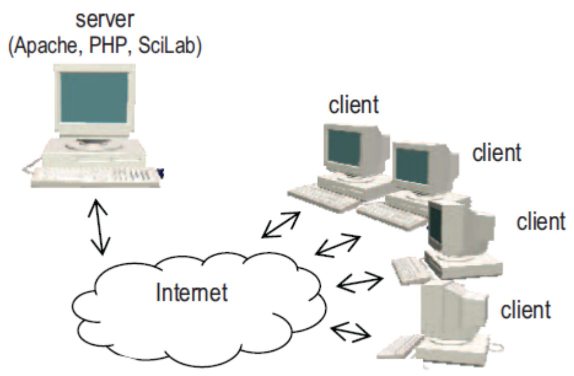

Figure 3. Communication between one server and several clients [3]

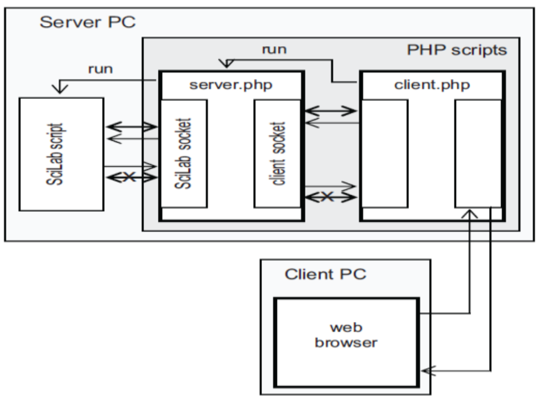

Figure 4.Technical realization of the socket communication [3]

III. SCILAB INTERFACE & CONTROL USING DIFFERENT DEVELOPMENT BOARD

The use of embedded processors has the potential of reducing the size and cost, increasing the reliability, and improving the performance of control system.

A. EP9315 processor development board [4]

The development board used in this work is based on the EP9315 processor from Cirrus Logic, as shown in Figure 5. The EP9315 [16] is a highly integrated system-on-chip processor for consumer and industrial electronic products. It features an advanced 200 MHz ARM920T processor design with a memory management unit, separate 16KB instruction cache, 16KB data cache, 64MB SDRAM, and 32MB flash memory. Linux, Windows CE and many other embedded operating systems are supported. The ARM920T has 32-bit microcontroller architecture, along with a five-stage pipeline, and is capable of delivering impressive performance at very low power. The key software packages used include Linux, TinyX, JWM, and Scilab/Scicos. The flexibility, scalability, reliability, and free nature of Linux have made it an increasingly popular platform for a large number of applications.

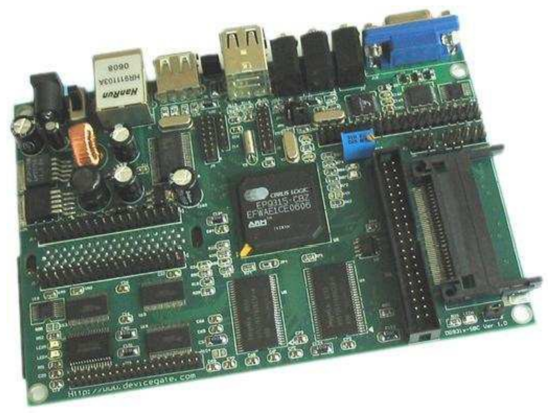

Figure5. EP9315 processor based development board

Scilab/Scicos was originally designed for PC-based systems but not embedded ARM-Linux systems. Therefore, it is necessary to port Scilab/Scicos onto the embedded platform.

Majority of core codes of Scilab are written in FORTRAN, so first a cross-compiler is built for g77 in order to support cross-compilation of GUI, for example. The GUI system of Scilab/Scicos is based on X11, and therefore the X11 server TinyX is included.

B. Arduino Galileo development board

Intel® Galileo is the first in a line of Arduino-compatible development boards based on Intel architecture. This development board will be made available from November 2013. [14]

The platform is built on the Arduino Galileo development board running a Linux operating system. The development board used in this work is based on the Quark SoC X1000. i.e. a single chip Pentium class system as shown in Figure 6. It is the heart of this unit.

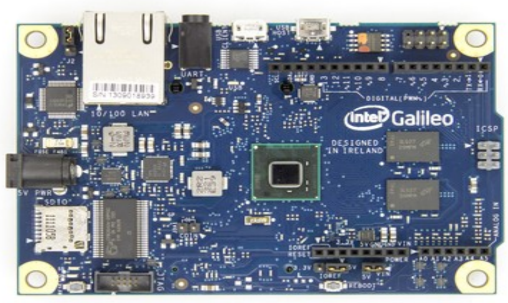
Figure 6 Intel Galileo development board

The Intel® Quark SoC X1000 Application Processor, is a 32-bit Intel Pentium-class system on a chip. It's the first board based on Intel® architecture designed to be hardware and software pin-compatible with Arduino shields designed for the Uno R3.ct. It also has the same I/O ports plus a full sized mini-PCI Express slot, 100Mb Ethernet port, Micro SD slot, RS232 serial port, USB host and client ports and 8MByte flash RAM. It also has a real time clock and a jumper that you can use to add a battery backup for it. [16]

Galileo runs Linux out of the box. It comes in 2 flavors, the default is a small Linux. By adding an SD card to the kit fully featured Linux can be added. Intel Galileo can work with any programming language that supports a .586 extension for x86 processors. Intel® Galileo currently runs on open source firmware based on C programming language. GCC and ICC compilers are supported. [15]

IV. CONCLUSION

It is possible to replace high cost propriety software with free open source software. Combining the strengths of different development boards & Scilab can offer new possibilities for application development. It is possible to interface and control system for an open source using different ways.

REFERENCES

[1] Liao Wenjiang, Dong Nanping and Fan Tongshun, "The Application of Scilab/Scicos in the lecture of Automatic Control Theory,"IEEE International Workshop on Open source Software for Scientific Computation (OSSC), pp. 85-87, September 2009

[2] Lilan Wu, ,Jianling Gao and Xiaoyao Xie, "Java Interface for Scilab Based on the JLab Environment," IEEE, International Conference on Anti counterfeiting, Security, and Identification in Communication, pp. 588-591, 20-22 Aug. 2009

[3] Zoltan Magyar and Katarina Zakova, "Using Scilab for Building of Virtual Lab," 9TH IEEE International Conference on Information Technology Based Higher Education and training (ITHET), pp. 280-283, April 2010,

[4] Longhua Ma, Feng Xia and Zhe Peng, "Integrated Design and implementation of Embedded Control Systems with Scilab,"K. Elissa, "Title of paper if known," unpublished.

[5] . Mayur jain, Sheetal Bhande, Aditya Chhatre, Amit Naik, Vishal Pande, Prafulla Patil, "Control System Design Using Open source Software," International Journal of Engineering Research and applications, 30 march 2012.

[6] T. Reveyrand, The SOCKET Toolbox for Scilab, http://www.reveyrand.fr/

[7] Stephen L. Campbell, Jean-Philippe Chancelier, and Ramine Nikoukhah, "Modelling and Simulation in Scilab/Scicos," Springer,2006

[8] Scilab [Online]: Available: http://www.scilab.org/.

[9] Scicos [Online]: Available: http:www.scicos.org.

[10] RTAI [Online].
Available: http://www.rtai.org;

[11] Comedi [Online].
Available: http://www.comedi.org/.

[12] Links to companies [Online].
Available: http://www.linuxdevices.com/ .

[13] Comedi/Comedilib Tutorial and User Manual "comedilib.pdf".
Available: http://www.comedi.org/.

[14] Link to companies [online]
Available: www.adafruit.com/blog

[15] https://communities.intel.com/message/207619

[16] http://arduino.cc/en/ArduinoCertified/IntelGalileo

[17] http://www.tutorialspoint.com/operating_system/os_linux.htm