Testing and Measurement Work bench in Real-Time Linux Environment

Chirag Panchal, Prof. V.P. Patel

Abstract— **this paper is concerned with possibility to develop real time platform with free open source code software. It describes the control ability of personal computer with Linux operating system, cooperation with free open source simulation & control software SCILAB/SCICOS, interfacing driver code COMEDI. The Real-time Application Interface (RTAI) freeware is used to create hard real time system clock. That make Linux kernel full preemptive and deterministic. The main advantage of this type platform is that all software or codes are freely available from internet website.**

Keywords **— Comedi, RTAI, scilab/scicos, Linux.**

I. INTRODUCTION

A real-time system is an integral part of any modern control and monitoring application systems. Normally, real-time control tasks are accomplished by employing dedicated processing units such as digital signal processor or advance micro controller unit [5]. Generally real-time control Environment is based on the commercial software MATLAB/Simulink/Real time-Workshop CACSD (Computer Aided Control System Design) software or LABVIEW. These are used to generate and compile codes for different targets. The main disadvantage of this solution is cost of the software [1] [4].

Real-time control via PC (personal Computer) has its advantages. It is mostly the ability of easy and fast modification of control software, the possibility of further data processing for visualization or interaction with other control parts. There are several kinds of real-time add-ons for free open source Linux in the field of real-time automation by PCs. One of the most widely known is RTAI (Real-Time Application Interface). RTL (Real time Linux) has developed in two ways: *(1) Well supported Commercial version called RT-Linux PRO (2) Non-Commercial version called RT-Linux.* Free is based on RTL PRO. The free version of RTL may be considered as an independent product because the programing environment is quite different. It is possible to create device driver and apply it as the connection between RT-Linux and a hardware device according to manufacturers' specifications. A simpler way is to use an already existing free open software package, COMEDI (Control and Measurement Device Interface) [7] [16]. It is possible to develop real time control loop in open source mathematical and simulation software SCILAB and Simulink tool SCICOS via COMEDI add-ons for RTL. This environment allows to quickly crating real-time controller for real plants by generating and computing the full control application directly from the Scicos scheme [1].

The most obvious advantage of the designed this type Real-time platform is that all software or codes are free and available in web.

In order to get real-time platform a standard kernel must be configured in a Linux base and before this configuration it will include the patching of Hardware Abstraction Layer (HAL) or Adaptive Domain Environment for Operating System (ADEOS) with kernel. After patching and configuration the kernel, installation of the RTAI package must be carried out including rtai-lab and comedi. After this whole process, a set of kernel module are created in the user specified directory. Loading these modules, the real-time functionality is obtained [1] [10].

The paper is structured as follow: software packages overview described in section II. Installation of software packages (RTAI, COMEDI and SCILAB/SCICOS) is described in section III. In Section IV, various method to obtain real-time environment platform. Section V represents the Conclusion.

II. SOFTWARE PACKAGES OVERVIEW

Description of free open source software packages required to develop Real-time platform for modern control system.

A. Scilab

Scilab/scicos as main development environment. Scilab is one of the several open source alternative to MATLAB [13]. Scilab is cross-platform numerical computational package and a high-level, numerically oriented programming language. It is providing a powerful open computing environment for engineering and scientific application. Scilab has a toolbox for modeling, simulating and aiding the design of hybrid control systems. It can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations, numerical optimization and modeling, simulation of explicit and implicit dynamical system and symbolic manipulations. Since 1994 it has been distributed freely along with the sour code via the internet. It is currently used in educational and industrial environments around the world.

Scilab include hundreds of mathematical function with the possibility to add interactively programs from various

Chirag Panchal, Instrumentation & control, GTU/ LDCE/ Ahmedabad, Gujrat, India.

Prof.V.P.Patel , Instrumentation & control, GTU/ LDCE/ Ahmedabad, Gujrat, India .

languages (FORTRAN, C, C++, JAVA). It has sophisticated data structures including lists, polynomials, rational functions, liner systems [2] [9]. *Figure 1.* Shows Scilab console and graphical windows.

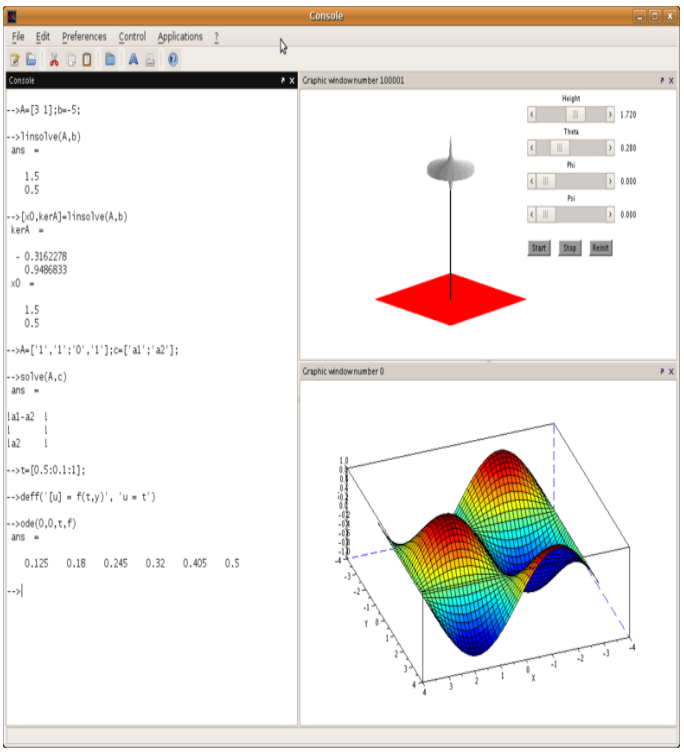

Figure 1. Scilab console [13].

B. Scicos

Scicos is a scicosLab toolbox for modeling and simulation of dynamical systems. Scicos is particularly useful for modeling systems where continuous-time and discrete-time components are interconnected. Such model can be programmed directly in Scilab using the *ode* and *dassl* function [9]. Scicos provides a modular way to construct complex dynamical systems using a block diagram editor. Within scicos graphical editor it is possible to place, configure and connect block, in order to create diagrams to model hybrid dynamical systems, and simulate it. Most of the scicos graphical user interface is written in Scilab language, for complete integration with Scilab, easy customization and maximum flexibility. Each blocks is represented by *two* functions [2]:

- The *interfacing function* written in Scilab language, define the graphical representation, the input, output and control ports and signals and the user configurable parameters;
- The *computational function* that is the real code used in the simulations. The computational function can be written in Scilab language, for easy development, or in C, for maximum efficiency and speed. The C computational function can be pre-compiled or "compiled-before-simulation" in order to allow the user to customize without leave the scicos environment.

Scicos simulations could be used to interact with real system in many ways [2]:

- Scicos HIL (Hardware In the Loop)
- The internal, general purpose, C generator. The internal "Code Generator".
- Use a specific code generator for Linux RTAI. This solution merge the hard real time capability of Linux-RTAI kernel with the creation of a standalone program from scicos computational functions, ideal for embedded systems with limited hardware resources.

Figure 2. Shows the scicos editor main windows and *Figure 3.* Shows the Palette browser in scicos. Figure 4. Shows the basic scicos diagrams.

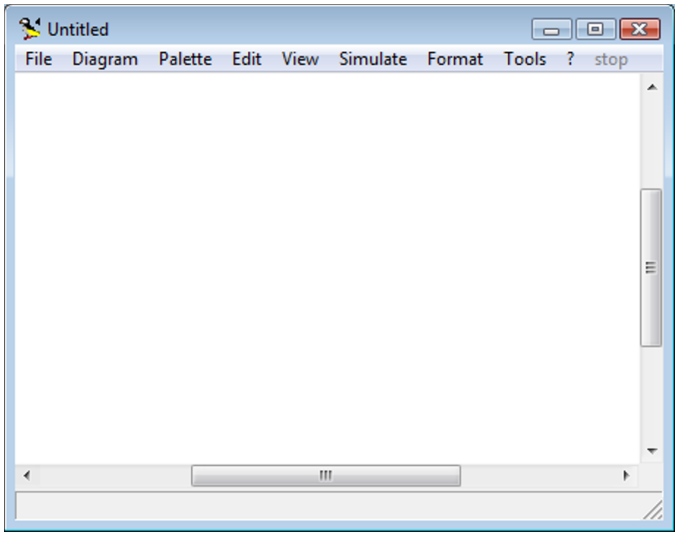

Figure 2. Scicos editor main windows [9].

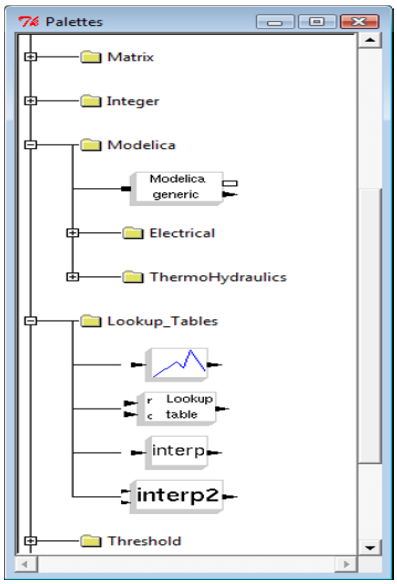

Figure 3. Palette browser [9].

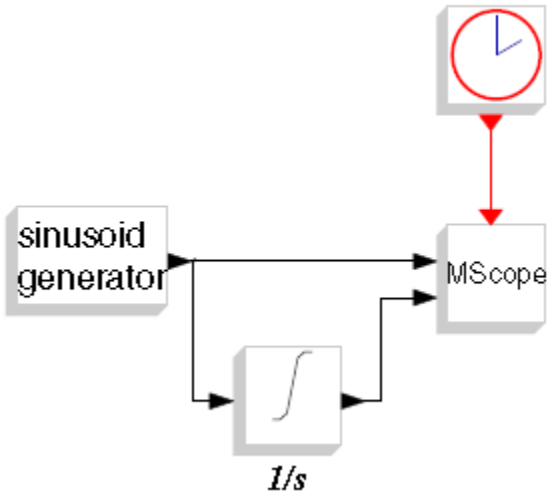

Figure 4. Scicos Diagram [9].

C. RTAI-Lab user interface

The basic concept of RTAI-Lab is to allow two separate systems, the host and the target, to communicate. In a remote implementation, the host is the machine where the RTAI-Lab graphical user interface (GUI) is executing in soft real time, the target is the machine where the generated hard real time code runs. RTAI-Lab that allow to display, in real time, the internal variable of a RTAI running controller, using scope and other virtual instruments. It is also possible to modify the internal parameter in order to tune the system. RTAI task and RTAI-Lab exchange data using IP address and NET-RPC protocol. The NET-RPC uses Ethernet (or other compatible media/interfaces) in real time using custom UDP data packet. This protocol support an arbitrary number to tasks/controller connected with a single supervisor. This is the ideal environment for distributed control applications. RTAI-Lab uses RTAI API to synchronize the communication and update the display with real time priority. This implies the presence of a Linux RTAI also in the supervisor PC that runs RTAI-Lab [2].

D. Comedi device drivers and libraries

Comedi is the standard way to use I/O interface devices with Linux. The Comedi project develops open-source drivers, tools, and libraries for data acquisition. Comedi is a collection of drivers for a variety of common data acquisition plug-in boards (more than 100 devices are supported). The drivers are implemented as a core Linux kernel module providing common functionality and individual low-level driver modules. To use the driver, the application program should use Comedilib functions. *Comedilib* is a user-space library that provides a developer-friendly interface to Comedi devices. Included in the Comedilib distribution is documentation, configuration and calibration utilities, and demonstration programs. *Kcomedilib* is a Linux kernel module (distributed with Comedi) that provides the same interface as Comedilib in kernel space, suitable for real-time tasks. It is effectively a "kernel library" for using Comedi from real-time tasks. This is a very important features because the hard real time RTAI tasks could not issues direct system call to uses [2].

E. Linux and RTAI-Linux kernels

The Linux kernel is the foundation over all this building is constructed. The Linux kernel provides a level of flexibility and reliability simply impossible to achieve with any other (free or commercial) operating system. For these reason many companies in the field of real time and embedded operating systems offer support for Linux.

The real time performances of the latest *2.4.xx Linux kernel* and *2.6.xx Linux kernel* are very close to real time [8] [10]. Linux is not a real time operating system because with the standard Linux kernel is not possible give guarantee about the absolute maximum latency of task activation. There are many project that modify (patch) the Linux kernel in order to create a "parallel" hard real time kernel (and user) space where the tasks have deterministic activation timing, while leave the other "normal" tasks running in the standard Linux user's space [2].

III. INSTALLATION

In the development of the low cost or open source real-time platform Environment, proper installation of prerequisite libraries packages and software must require.

A. Required Hardware and software or Codes

Personal Computer (PC) with Pentium or equivalent processor – 512 RAM. *Operating system:* Functional GNU/Linux environment, better with a Debian or a Debian-like (e.g. Ubuntu) distribution. *A kernel*: it is necessary to ensure the best when the kernel version of the Linux-OS is as close as possible to the kernel we are going to compile and to merge with the RTAI. Require RTAI source code, Scilab/Scicos source code, and COMEDI & COMEDI-LIB source. Another two supporting source codes are required to install. First one is *"Mesa 3D"* graphical library from and second one is the *"EFLTK"* graphic widgets library [1] [8].

Some software packages may have to upgrade or install and those are Automake>1.7, autoconf, bison (for comedi) cpp, ftgl-dev (for efltk), gcc, g77, g++, gtk, libbind, libglu1-mesa-dev, libglut-dev, libfltk, libgtk-dev, libdrm-dev, libncurses, libperl-dev,mesa (related all packages), libtool, doxygen, tcl8.4, tk8.4, tcl-8.4-dev, tk8.4-dev, tcllib-1.9, x11-proto, gettext, flex, and svn & cvn [8] [12].

B. Installation process

1. Install Operating system Ubuntu 12.04 LTS in PC.
2. A real-time task in RTAI is implemented as a kernel module which is loaded into the kernel after the required RTAI core modules have been loaded. This architecture yields a simple and easily maintainable system that allows dynamic insertion/ removal of the desired real-time capabilities and tasks. The steps followed to install RTAI are briefed below [5]:

- Select the version of RTAI, the Linux kernel version to use, and a Linux distribution that uses that kernel. One can choose them in any order, but it is important to end up using a consistent set. For example, with Ubuntu Linux distribution, to install RTAI 3.5, first need to install Ubuntu 12.04 distribution which has Linux Kernel version 3.5.0. A lower version of kernel 2.4.30 is chosen for which the RTAI patch is available [17].
- Install the Linux distribution.
- Download a clean version of the Linux kernel from the Linux distribution sites.
- Download the tar file for the RTAI release.
- Unpacking and uncompressing the files.
- Using the patch file in the RTAI release that corresponds to the kernel source you have downloaded, patch the kernel source to incorporate the RTAI modifications.
- Generate a configuration file that suits the machine you are going to run on.
- Build and install the Linux kernel.
- Configure, build and install RTAI.
- Check for consistency of the installed files.

3. Configuring the kernel for real time applications.
4. Compilation and Installation of the newly configured kernel.
5. Updating of the boot loader to access newly installed kernel.
6. Mesa and EFLTK installation
7. Installation of COMEDI and COMEDI-LIB [18].
8. Configuration, compilation and Installation of RTAI.
9. Installation of Scilab & RTAI add-on to it [13] [14].
10. Loading RTAI, COMEDI and DAQ modules [1].

IV. METHODOLODY

There are several method to develop Real-time platform for real-time control system. Scicos with RTAI add-ons is used to interface real-time system. Mostly used configuration are HIL, direct C code generation for scicos and specific code generator for Linux RTAI.

A. HIL configuration [3]

Real-Time Hardware-In-the-Loop (HIL) Simulation is increasingly recognized as an essential tool for engineering design. Real-time HIL simulation replaces the emulated hardware under test or control logic in the simulation model with real hardware that interacts with the computer models. This solution increases the realism of the simulation and provides access to hardware features currently not available in software-only simulation models. There is possibility of executing Real-time HIL simulation by using the Virtual Test Bed (VTB). The structure of the Real-time VTB & HIL simulation system is illustrated as *Figure 5.*

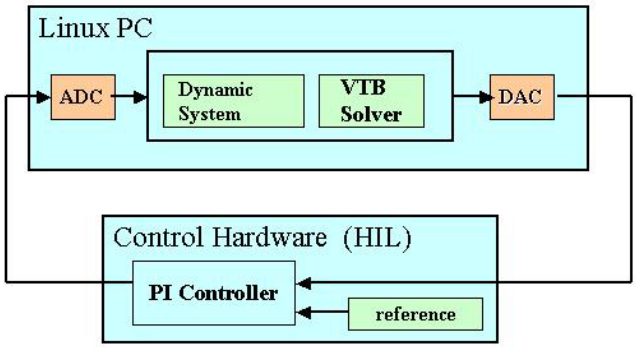

Figure 5. Structure of Real-time HIL Simulation System [3].

The Virtual Test Bed (VTB) is a software environment that has been developed for design, analysis, and virtual prototyping of large-scale multi-technical systems. The two most important features of the VTB are(i) it has the capability of integrating models that have been created in a variety of languages such as SPICE (Simulation Program with Integrated Circuit Emphasis), ACSL (Advanced Continuous Simulation Language) and Matlab/Simulink, into one simulation environment; and (ii) it provides advanced displays of simulation results including full-motion animation of mechanical components, oscilloscope-like displays of waveform data, and imaginative mappings of computed results onto the system topology.

Figure 6. Shows the whole process of Real-time implementation in RTVTB.

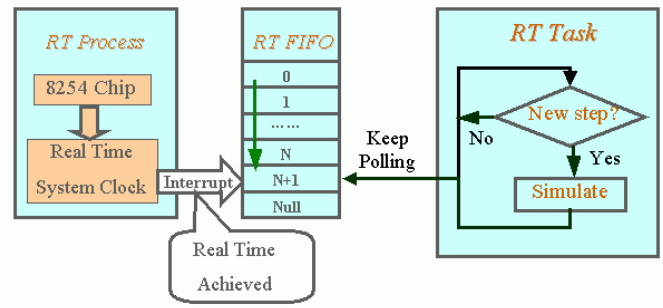

Figure 6. Real-time Implementation in RTVTB [3].

B. C code generation for scicos [4]

A modified Scicos code generator directly generates the C-code for the Linux RTAI environment. The user can perform all phases of control system design (see *Figure 7.*) within a unique environment.

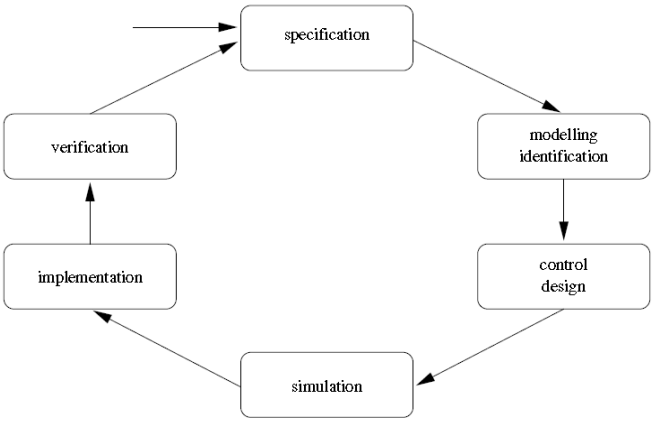

Figure 7. Control system design phases [4].

The RTAI environment has been extended with the new code generator *RTAICodegen_sci*. This is a modified version of the *Code Generation_sci* file provided by Scicos. The most important modifications are:

- Only the files for *standalone* execution are generated. The generation of the dynamic library for Scicos has been eliminated in the new code generator.
- The generated *Makefile* has been adapted in order to produce code for Linux RTAI.
- The functions *make_actuator* and *make_sensors* produce a more detailed code for each input and output of the superblock to allow easy integration of custom code. Input and output blocks should be normally implemented within the superblock.
- The function *make_static_standalone* creates an array with the names of the Scicos blocks found in the field identification. These names are used to univocally identify the block parameters in the GUI applications.
- The RTAI stand-alone executable code is generated and compiled directly from the Scicos environment.

The directory .../macros/RTAI contains a new block library specific for the RTAI environment. *Figure 8.* Shows the blocks contained in this library.

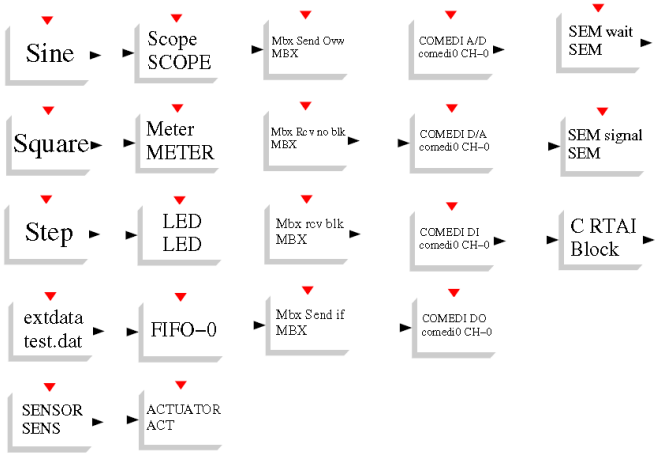

Figure 8. Scicos RTAI library [4].

Figure 9. shows the compilation process needed to obtain the RTAI stand-alone executable code.

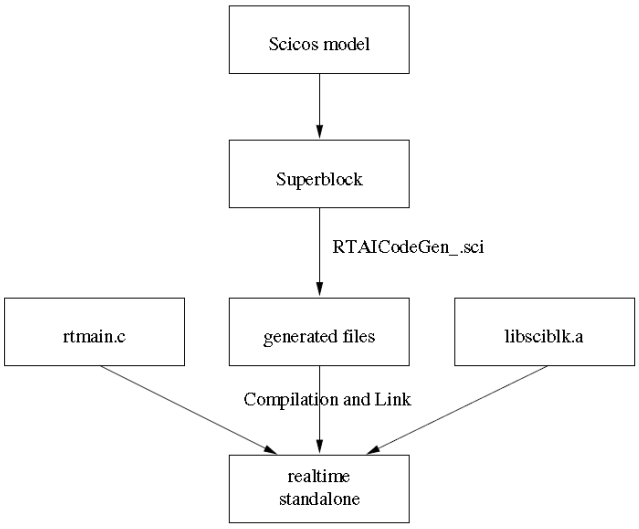

Figure 9. Code generation and creation of the stand-alone executable [4].

C. *Other Configuration [1]*

if HIL or direct C code generation method is not used to create Real-time platform than other option is to develop real-time platform by direct COMEDI block as a add-ons of scicos is used to interface real-time control system (see *figure 10*).

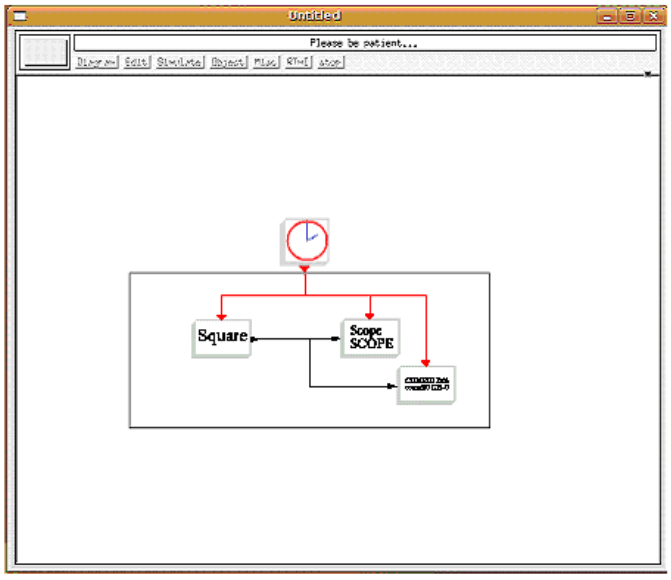

Figure 10. Making of RTAI Super block [1]

Direct control is possible via COMEDI tool box in scicos.

V. CONCLUSION

It is possible to replace high cost propriety software with free open source software.

It is possible to develop low cost Real-time platform Environment using free open source software code SCILAB/SCICOS, COMEDI, and RTAI in open source Linux operating system. This Environment can be used to control

real-time application and also used for testing of hardware system.

REFERENCES

[1] Ujjwal Mondal, Parthasarathi Satvaya, Sourav Kumar Das, "Real-Time Speed Control of a DC Motor using Open Source Code Tools", International Journal of Soft Computing and Engineering (IJSCE) ,ISSN: 2231-2307, Volume-2, Issue-6, January, 2013.

[2] Simone Mannori , Ramine Nikoukhah , Serge Steer, "Free and Open Source Software for Industrial Process Control Systems", INRIA-Rocquencourt, Domaine de Voluceau, 78153 Le Chesnay Cedex, France.

[3] B. Lu, W. McKay, S. Lentijo, A. Monti, X. Wu, and R. Dougal, "The Real Time Extension Of The Virtual Test Bed", Department of Electrical Engineering University of South Carolina Columbia, SC 29208, USA

[4] Roberto Bucher and Silvano Balemi, "Scilab/Scicos and Linux RTAI - A unified approach" , IEEE Conference on Control Applications Toronto, Canada, August 28-31, 2005.

[5] Pragyan Pratim Hazarika and K. N. Shubhanga, " Development of a Relay Test Bench and an Arbitrary Waveform Generator in RTAI-Linux Platform" , IEEE ICECCN 2013

[6] Luis Garcia, Manuel J. Lopez, Jose Lorenzo, "Hardware-in-the-loop Environment for Control Systems evaluation under Linux/RTAI", International Conference on Applied Computer Science, Tenerife, Canary Islands, Spain, December 16-18, 2006.

[7] Matej Dobsovic, Martin Kratmiiller, "Control an Electromechanical system in the Real-time Linux Environment", Acta Polytechnical Hungarica, Vol.5,No. 2,2008.

[8] Roberto Bucher, Simone Mannori, Thomas Netter , "RTAI-Lab tutorial: Scilab, Comedi, and real-time control" , February 28, 2008.

[9] S. L. Campbell, J.-P. Chancelier, and R. Nikoukhah. "Modeling and Simulation in Scilab/Scicos with ScicosLab 4.4. Springer", New York.

[10] Giovanni Racciu and Paolo Mantegazza. RTAI 3.3 User Manual, 2006. URL www.rtai.org.

[11] R. Bucher and L. Dozio, "CACSD with Linux RTAI and RTAI-Lab," in Real Time Linux Workshop, Valencia, 2003.

[12] Mantegazza, y S. Papacharalambous, "Real time application interface", Linux Journal (2000).

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

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

[15] RTAI [Online].
Available: http://www.rtai.org;
Available: http://www.dti.supsi.ch/ bucher/scilab-howto.pdf.

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

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

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

Chirag Panchal, Instrumentation & control, GTU/ LDCE/ Ahmedabad, Gujrat, India,

Prof.V.P.Patel,Instrumentation & control, GTU/ LDCE/ Ahmedabad, Gujrat, India .