UKACC International Conference on Control 2012
Cardiff, UK, 3-5 September 2012

Design and Implementation of Real-Time Control System Using RTAI and Matlab/RTW

An Baoran, Guoping Liu
Center for Control Theory and Guidance Technology
Harbin Institute of Technology
Harbin, China
baoranan@gmail.com, gpliu@hit.edu.cn

Chai Senchun
School of Automation
Beijing Institute of Technology
Beijing, China
chaisc97@hotmail.com

Abstract—**In this paper, Linux-RTAI and Matlab/RTW is used to develop real-time control system for fast real-time simulation and implementation of the control algorithm. The hardware of the controller is based on PC/104 cards under Linux-RTAI operating system. The main characteristic of Matlab/RTW is that it can automatically generate the real-time simulation code for many target processors. The combination of Linux-RTAI as a controller platform and Matlab/RTW as control algorithm development environment brings the two main advantages: hard real-time capability and reduced time for control development. Moreover, online signal monitor and parameter modification is realized on the whole structure system. At last, the minimum execution time of real-time control task is determined and the application of real-time control system into flight vehicle is achieved in the work.**

Keywords-RTAI; real-time; Matlab/RTW; PC/104; controller

I. INTRODUCTION

With unceasing consummation and development of control theory and computer technology in the field of automation, real-time capability is the important guarantee to realize the designed control algorithm. Real-time systems are those that can support the execution of applications with time constraints [1]. Basically, real-time systems are classified into two categories: hard real-time and soft real-time. In a hard real-time system, deadlines cannot be missed and the success depends on the task execution within deadlines [2][3][4]. Whereas in the soft real-time systems, there is a certain degree of tolerance for missing deadlines[5]. The importance of the deadlines depends on the type of control system.

Hard real-time control requires a software platform that guarantees certain time constraints. Real-Time Operating System(RTOS) should meet the basic requirements including predictability, pre-emptability, support for multi-threaded scheduling, concept of priorities, and resource sharing mechanisms avoiding priority inversion. Today, there are many available RTOS applied on the market, such as VxWorks[6],uCos[7],QNX[8] and Windows CE[9],etc., while the RTOS based on Linux are gaining attractions due to its robustness and open source code. And researchers, developers, and programmers prefer Linux as the open platform when developing applications. Based on the Linux kernel, RTAI was developed as a real-time operating environment solution at Dipartimento di Ingeneria Aerospaziale Politecnico di Milano (DIAPM) [10].It extends the Linux kernel with hard real-time functionality in addition to allowing the use of all standard Linux drivers and applications. Compared to the commercially available real-time OSs, The RTAIs performance is very competitive with the best commercial RTOSs. RTAI is open source and free under the terms of the GNU. So all needed software can be freely downloaded on the web during the building of the real-time control system.

Moreover, the diversity of control algorithms and development cycle of the projects are very important for engineers and researchers to develop and design every control project. Matlab, which stands for Matrix Laboratory, is an easy-to-use and powerful software for simulation aid design and technical computing. It not only provides Matlab language but also other application program interface to programming languages such as C, C++ and Fortran. Simulink is a graphical toolkit for modeling and simulation which can carry out many simulations of many aspects such as electronics, control engineering and signal processing. The Real-Time Workshop (RTW) provides a C code generator environment for rapid prototyping and development [11][12][13]. It generates source code from Simulink models to create real-time applications, which can operate in PC, ARM, DSP, PC/104, PLC and other hardware systems[14].

Therefore, combining Linux/RTAI and Matlab/RTW, one real control system is established in our project in order to realize hard real-time capability and a rapid development.

The remainder of this paper is organized as follows. The next section presents structural description of the whole real system. Section 3 presents the process of designing real-time controller. Section 4 presents software configuration and development on Windows PC which can implement automatic code generation and transplant into control application. Experimental results and conclusion are given in last two sections.

II. SYSTEM ARCHITECTURE

The whole system structure includes mainly two parts, one is the real-time control system comprised by hardware and software which can provide the run-environment for real-time task execution, the other is the virtual simulation environment composed of computers and related software. The system structure is shown in Figure 1. PC is the core of the design of control block diagrams and automatic generation and compilation for C code, which have Matlab, Simulink, virtual machine and other related software environment. The objective of PC focuses on how to produce and download executable program applicable to the target computer (real-time control system). The target platform is the RTAI target run-time environment. It uses embedded industrial controlling computer PC/104 as real-time controller, which is responsible for receiving executable code from PC by TCP/IP protocol and data processing of various parameters under real-time environment Linux/RTAI. According to data processing results, it can make decision and produce control information to control the running state of controlled object.

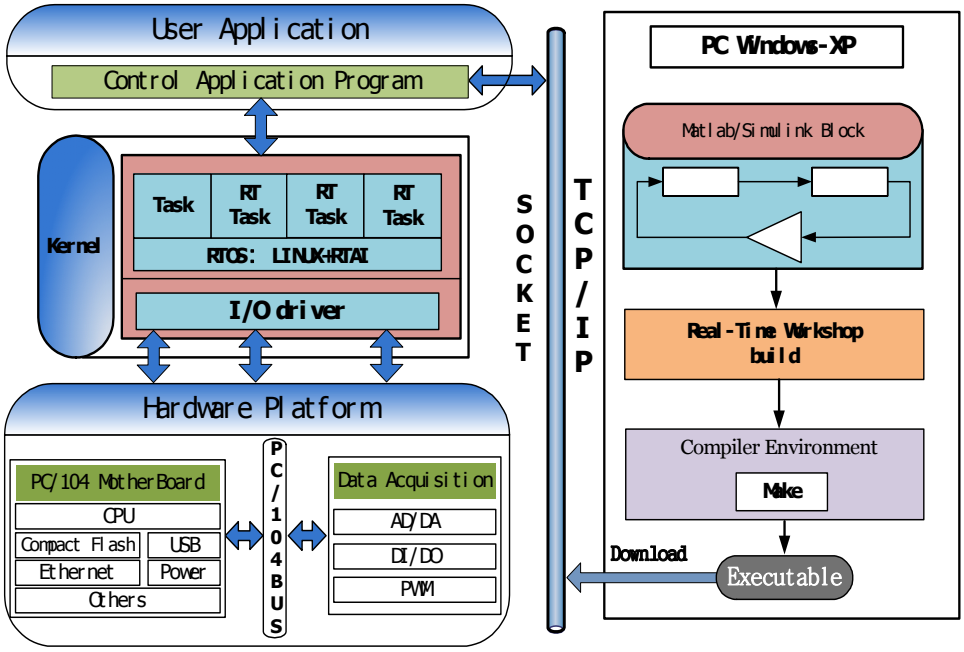


Figure 1. Overall structure

III. Real-Time Control Target Platform

According to complexity and real-time demands of various kinds of controlled plants, the real controller is developed based on industrial modular design methodology in the following.

A. Hardware System

In order to meet the requirement of a hard real-time environment, boards compliant to the PC/104 standard are chosen as the basis for real-time control system. PC/104 is an industrial control bus, which has some unique features such as small size (96 mm × 90 mm), stack-type connections, low bus drive current and so on[15]. The following two cards utilized in the Linux/RTAI system incorporate the PC/104 bus.

The heart of the hardware system is the CPU card. A PMI2 [16] card manufactured by SBS Science & Technology Co.,Ltd. was selected. The highly integrated CPU board is equipped with:

- Intel Pentium M running at 2 GHz;
- 256 MB of RAM;
- four USB 2.0 ports;
- two serial ports and one parallel port;
- industrial Ethernet interface;
- a CF card socket onboard;
- PC/104 bus.

A PC/104 ADT652 I/O module [17]is added to the stack to implement data acquisition and signal output. The hardware configuration of the card is equipped with the following components:

- 16 analog inputs, 12-bit A/D resolution;
- 100KHz A/D sampling rate;
- 4 analog outputs, 12-bit resolution;
- 24 digital I/O with programmable direction;
- 4 16-bit general purpose PWM ports.

B. Software Design

An important item that allows the use of PC/104 cards in the application of various control algorithms is a real time operating system Linux/RTAI. In the design of the software system, it should include root filesystem, kernel image, TCP

/IP network support, necessary tools and I/O interface drivers. The design process of system software is given below.

1) Installation GRUB bootloader.

GRUB is the first software program that runs when a computer starts. It is responsible for loading and transferring control to the operating system kernel software. By using the Grub loader, it is very convenient for booting a GNU/Linux system directly from a CF disk.

2) Linux 2.6 kernel configuration and filesystem transplantation.

The new Linux v2.6 offers a host of improvement in the configuration and real-time features compared with the earlier Linux kernel. Therefore, the design chooses Linux 2.6.23 kernel version.

In order to make Linux real-time, the kernel source was patched with the correspondent RTAI patch. In the paper, RTAI version is 3.8 which includes hal-linux-2.6.23-i386-1.12-03.patch for making Linux 2.6.23 real-time. "HOWTO install RTAI"[18] describes instruction on how to configure the Linux kernel and RTAI target in detail.

Building the root filesystem with Busybox involves selecting files necessary for the system to run. Here is a reasonable minimum set of directories for the root filesystem(ext3): /bin, /boot, /dev, /etc, /home, /lib, /mnt, /proc, /sbin, /usr. These directories and files are copied into CF card in reference to "Building a root filesystem" [19].

3) ADT652 I/O card driver modules including AD, DA, DI, DO, PWM.

A device driver's function is to provide access to a piece of hardware and the Linux Kernel provides a stable user space interface or applications. In Linux, device drivers are created as modules of the Linux Kernel that hide the details of how the device works. Each module can be used or removed from the kernel at runtime depending on what hardware is available.

In order to realize data collection control, device drivers for ADT652 card are responsible for analog input signals collection, digital input signals collection, analog output, digital output and PWM signal output. Five modules for ADT652 I/O signals control are designed in C programming, which can be added into kernel dynamically.

4) Networked receive programs

In networked control environment, network configuration and programming for receiving executable files is the important guarantee for target platform to realize communication with the host computer (PC side).

There are three important executable files in the /bin directory for network communication: *inetd*, *telnetd* and *recvn*. The *inetd* daemon is a super-sever daemon that manages Internet services on Linux system and the *telnetd* daemon is a server that supports the DARPA-standard TELNET virtual-terminal protocol. They are started automatically as the system starts up so that it is more convenient for users to login Linux/RTAI system and have remote control and on-line modifying from PC side to target computer.

Networked receive process named *recvn* is designed for receiving executable files generated from Simulink block diagrams by RTW in PC computer via TCP/IP using Socket. After successful receipt, the process creates a child process which executes the generated executable.

Figure 2 below illustrates the program execution flow after the target system starts.

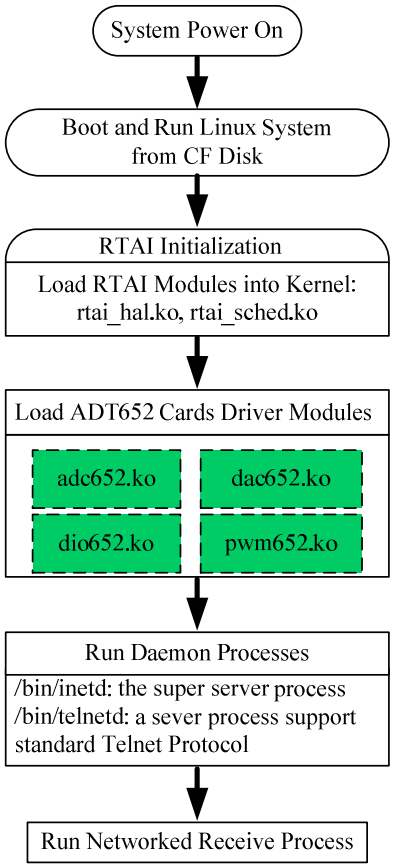


Figure 2. The execution of system program

IV. SOFTWARE CONFIGURATION AND DEVELOPMENT OF HOST SYSTEM

In this section, host PC's software configuration and development is covered in details. Figure 3 describes the functions of software on host PC when a control project is built. Matlab and VMware Workstation are two necessary software components for the host PC (Windows XP Computer). As mentioned above, Matlab is responsible for constructing the control block diagrams in Simulink and generating C code through RTW. VMware Workstation, which is a kind of virtual machine software, can run various operating system images at the same time. A real-time Linux operating system is installed in VMware Workstation so that it can provide compile environment for converting generated C code into executable file. In host PC, another software named 'NetConTop' is developed to have online monitoring of parameters and signals of executable program running on PC/104 target. For details, see [20].Here is the main development process of Matlab/Simulink/RTW and VMware RT-Linux-OS.

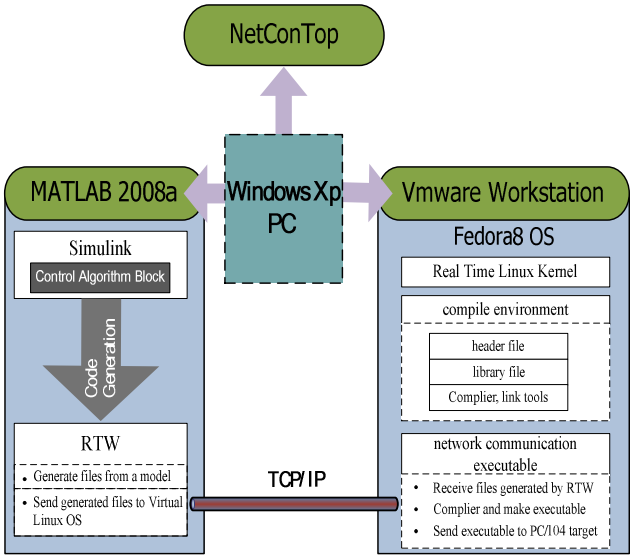


Figure 3. The structure of system software

A. Matlab/Simulink/RTW

1) I/O Blocks Development

I/O blocks are necessary elements from which real-time models can be built. Simulink enables designers to create custom blocks with C-MEX S-function. C-MEX S-function, which is written in C programming language and compiled as MEX-files, defines block properties and allows users to add desired blocks to Simulink models. In the work, I/O blocks are configured through custom blocks and listed below (Table 1).

TABLE I. I/O BLOCKS

Block Name	Parameter	Port Number	
		Input	*Output*
ADC	● Input Channel ● Sample Time	1	0
DAC	● Output Channel ● Sample Time	0	1
DI	● Input Channel ● Sample Time	8	0
DO	● Output Channel ● Sample Time	0	8
PWM	● Output Channel ● Frequency ● Sample Time	0	1

2) Library Files for Code Generation

The process of generating target-specific code is mainly controlled by a target language compiler file and a template makefile. To support Linux/RTAI environment, the target language compiler file named '*grt_pc104.tlc*' is developed to control the way code is generated by RTW and the template makefile named '*grt_pc104.tmf*' is created to generate one right makefile for compile environment on Linux platform.

3) Main Program of Model Files

To obtain real-time model executable, the generated code should be compiled and linked with specific library to one main program. At this point, a main file named '*rtmain.c*' is developed. It includes a main thread and two slave threads. When the executable starts, the main thread initiates the other two slave threads: one starts the real-time control task, the other is responsible for the communication between the real-time control task and the external GUI monitoring software (NetConTop).

4) Files Sending Program

The generated code should be compiled on Linux platform, but Windows is much more popular with most PC users. So Virtual Machine is introduced to create virtual compile environment (Section B will give a detailed account of the process). In this case, a file-sending program named '*sendfilewin.exe*' is designed to send the generated code to Virtual Machine based on Windows Sockets. The command of calling this program is added into '*grt104.tmf*' so that the generated code can be sent automatically by a mouse click.

B. VMware/RT-Linux-OS

1) Build Compile Environment

VMware Workstation, which is a kind of virtual machine software, can run various operating system images at the same time and get connections to its host or other computers on the network after the right network configurations. Fedora 8, one of the most popular free-as-in-freedom Linux distributions, is installed on VMware Workstation 7.1 running on Windows XP PC host. In order to build complete compile environment, RT-Linux OS is created by using the same methodology as that used in constructing a real-time Linux kernel based on the above Fedora-VMware platform. As necessary files including header files, library files are added into the virtual RT-Linux environment by the way of file sharing, Fedora-VMware system has the ability of compiling and linking the generated code to the final executable file which has the same name as the model file.

2) Network Communication Program

Figure 4 shows data communications among the three components: Matlab, Fedora-VMware and PC/104 target. In Fedora-VMware environment, one network communication program named '*recvfdr*' is designed by using Linux sockets. It has functions in the following:

a) receive the generated files sent by sendfilewin.exe;

b) compile and link main file, generated files and library files together into a single executable file model;

c) download the executable model to PC/104 target.

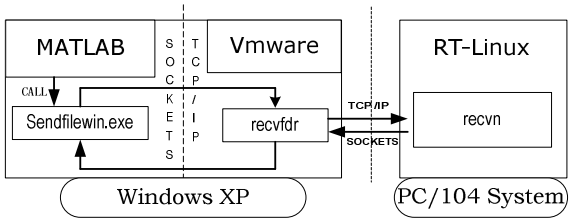


Figure 4. The structure of network communication

V. APPLICATION EXAMPLE

The following examples illustrate the process during the implementation of a PID closed loop control for a four-wing

flight simulator in order to have a hardware-in-the-loop test of one designed real-time control system.

A. Matlab Toolbox Testing

The Matlab version in test is 2008a, and contains totally 804 blocks including Simulink, Control System Toolbox, Fuzzy Logic Toolbox and so on. After the test, there are only 96 blocks failed in the test, and the rest of which can be used in the system well.

B. Real-Time Performance Evaluation

In order to get the minimum execution time of real-time control task, the blocks of DAC and PWM are tested to evaluate the system's real-time performance, and finally the minimum sample period is determined as 18us. As for the medium scale programs, the control task can be finished within one minimum period of 500us.

C. Fight-Attitude-Control Experiment

Figure 5 shows the construction of four-fixed-wing flight simulator system. Two couples of DC Motors as well as gears are fixed on both ends of the beam to drive respective propellers. The motion of propellers can rotate the flight simulator in pitch or yaw. For example, a couple of horizontal propellers can generate a vertical force, causing the flight simulator to pitch up or down. Pitch angle and yaw angle are two important flight dynamics parameters that can be detected by two angle displacement sensors mounted on the pivot and sent to controller unit as the feedback information. Based on the feedback of pitch and yaw angles, the desired controller perform a control signal to motors through a power-expansion circuit in order to realize the attitude control of the flight simulator.

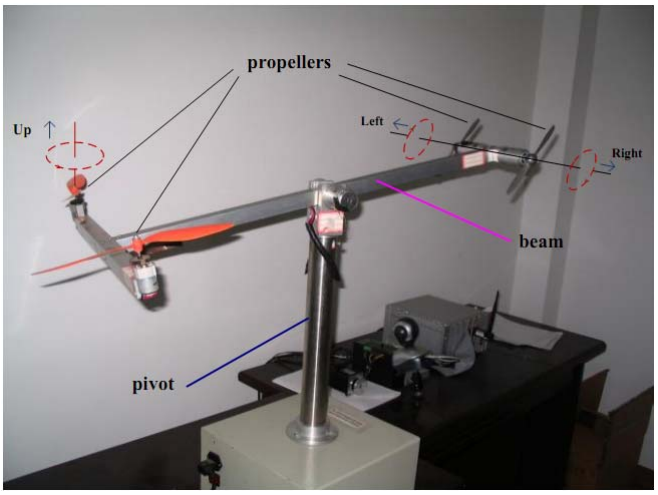


Figure 5. Four-fixed-wing flight simulator

The design and implementation fight-attitude-control system is shown in the following steps.

Step 1 Design a model file including ADC blocks, DAC blocks and other system blocks for both controller and plant in Simulink. In the paper, a PID controller with Loop-up compensator is designed for flight attitude control implementation as shown in Figure 6.

Step 2 Configure IP addresses of VMware and PC/104 target in Matlab/RTW dialog box and start automatic compile and download process by clicking build button.

Step 3 Use NetConTop software to have a monitor of PC/104 controller. In the project of NetConTop, the signals and parameters of the control model can be obtained and tuned with GUI application by users.

Finally, experimental results showed that the entire controller could complete all its tasks within the sampling period of 400 μs. Figure 6 shows the control block diagram and the monitoring interface of flight simulator system.

VI. CONCLUSION

In order to satisfy the requirement of fast development and hard real-time capability, the paper takes Matlab/RTW and RTAI as software combination, and it realizes the modules imaging, modularization and configuration. The process of hardware and software design for real-time control system is described in details. At last, experimental results show that one control block diagram can complete all of its tasks within the sampling period of 500us and the application into flight vehicle is achieved.

ACKNOWLEDGMENT

The authors would like to thank the referees for their valuable and helpful comments which have improved the presentation.

REFERENCES

[1] Ramamritham Krithi, Stankovic John, “Scheduling Algorithms and Operating Systems Support for Real-time Systems,” Proceedings of the IEEE,1994,82(01) doi:10.1109/5.259426

[2] L. Sha, T. Abdelzaher, K. E. Arzen, A. Cervin, T. P. Baker, A. Burns, G. Buttazzo, M. Caccamo,J. Lehoczky, and A. K. Mok, “Real time scheduling theory: A historical perspective,” Real-Time Systems,vol.28, no.4, pp.101-155, Nov. 2004.

[3] Giorgio Buttazzo, “Hard Real-Time Computing Systems: Predictable Scheduling Algorithms And Applications,” Second Edition, Springer, 2005.

[4] E. Douglas Jensen. “Hard and soft real-time,” http://www.real-time.org/,2011.

[5] C. Lin and S. A. Brandt, “Improving Soft Real-Time Performance Through Better Slack Management,” Proceedings of the IEEE Real-Time Systems Symposium (RTSS 2005), pp.314, Miami, Florida, December 58, 2005.

[6] VxWorks, Available: http://www.windriver.com/vxworks

[7] uCos, Available: http://www.micrium.com/

[8] QNX, Available: http://www.gnx.com/

[9] Windows CE, Available: http://www.microsoft.com/windows/embedded/wince

[10] Dipartimento di Ingegneria Aerospaziable Politecnico di Milano.RTAI Homepage. http://www.rtai.org/,2010.

[11] Mathworks Inc. Real-Time Workshop. http://www.mathworks.com/products/rtw/,2010.

[12] Bucher, R., Balemi, S., “Rapid Controller Prototype with Matlab/Simulink and Linux,”Control Engineering Pracrice 14, 2003,pp.185–192.

[13] PC/104 Specification [M]. Version 2.5. PC/104 Embedded Consortium, 2003.

[14] Gherasim C.,Van den Keybus, J.,Driesen, J.,Belmans, R.,"DSP implementation of power measurements according to the IEEE trial-use standard 1459," IEEE Trans on Instrumentation and Measurement Vol. 52,no.4,pp. 1086-1092,2004.

[15] http://www.sbs.com.cn/en/productshow.asp?id=740

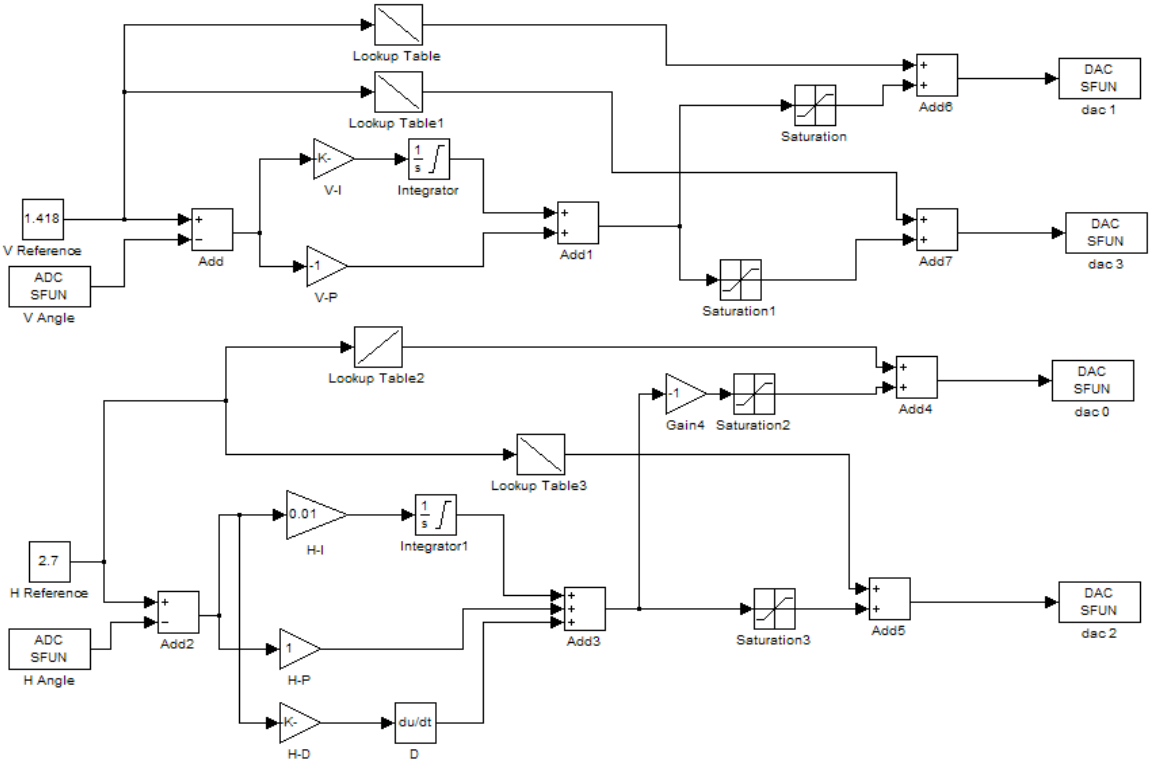
[16] http://www.sbs.com.cn/en/productshow.asp?id=692

[17] Dozio, L.,Mantegazza, P., "Real Time Distributed Control Systems Using RTAI," Proc. IEEE Symp. Object-Oriented Real-Time Distributed Computing, IEEE Press, May 2003, pp.11-18, doi: 10.1109/ISORC.2003.1199229

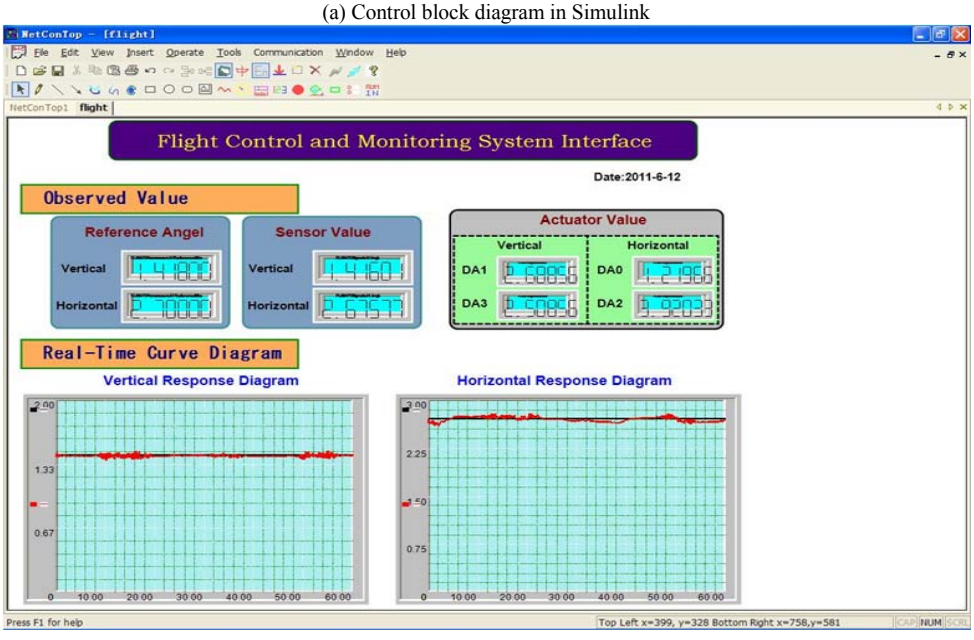
[18] http://www.isr.uc.pt/~rui/str/howto_install_rtai.html

[19] http://tldp.org/HOWTO/Bootdisk-HOWTO/buildroot.html

[20] PANG Zhonghua, LIU Guoping, Zheng Geng, Dong Zhe. Rapid Realization of Networked Control Systems Based on NetCon[J].Chinese Journal of Control and Instruments in Chemical Industry.2009,36(5):79～83.



(a) Control block diagram in Simulink



(b) Monitoring interface based on NetConTop

Figure 6. Experimental result of the fight-attitude-control system