ISSN:1306-3111
e-Journal of New World Sciences Academy
2008, Volume: 3, Number: 1
Article Number: A0050

NATURAL AND APPLIED SCIENCES
ELECTRICITY ENGINEERING
Received: July 2007
Accepted: December 2007

Mustafa Onat
University of Marmara
monat@marmara.edu.tr
Istanbul-Turkiye

DEVELOPING REAL-TIME CONTROL EDUCATION APPLICATIONS INTERACTING WITH THE MATHWORKS ENVIRONMENT

ABSTRACT

This study serves as a guide on developing real-time prototyping control applications interacting with MathWorks environment. In this platform, students can easily design controllers in real-time, implement their own algorithms, and obtain graphical results without requiring special programming skills. Considering its widely usage in practice and theory, developing the real-time speed control of a PM DC motor is introduced as a case study.

Keywords: Rapid Prototyping, Real-Time Control, Simulink, xPC Target

GERÇEK ZAMANLI KONTROL EĞİTİMİ UYGULAMALARININ MATHWORKS ORTAMIYLA ETKİLEŞİMLİ GELİŞTİRİLMESİ

ÖZET

Bu çalışma, gerçek zamanlı prototip kontrol uygulamalarını MathWorks ortamı ile etkileşimli geliştirmede kılavuz niteliğindedir. Bu platformda, öğrenciler kontrolörleri kolayca gerçek zamanda tasarlayabilir, kendi algoritmalarını gerçekleştirebilir ve grafiksel sonuçlarını özel bir programlama becerisi gerektirmeden kolayca alabilir. Pratik ve teoride geniş kullanım alanına sahip olduğu düşünülerek bir PM DC motorun gerçek zamanlı kontrolünü geliştirme bir örnek çalışma olarak tanıtılmıştır.

Anahtar Kelimeler: Hızlı Prototipleme, Gerçek Zamanlı Kontrol, Simulink, Hedef xPC

1. INTRODUCTION (GİRİŞ)

MathWorks' rapid prototyping software presents many facilities such as automatic coding, hardware in the loop testing and getting rapid graphical result, which becomes more wide spread amongst academia and industry [1].

This paper presents the teaching of xPC Target for general propose real-time applications with a case study.

A real-time control system consisting of hardware components (PC and DAQ card) and software tools (Simulink, xPC Target, RTW and C++ compiler) is implemented and experimented with for teaching purposes.

The similar purposed studies [2 and 3] using the MathWorks' environment in real-time implementations are studied for design and implementation purposes. This study is focused solely on teaching the real-time implementation and guiding the students on their lab sessions.

The paper starts by describing the rapid prototyping using a MATLAB/Simulink environment and its implementation. Next, the controller together with its design structure and its implementation are explained. Then the work continues describing the experimental setup. Experimental results are presented highlighting features of rapid prototyping control. Finally, concluding remarks are provided.

2. RESEARCH SIGNIFICANCE (ÇALIŞMANIN ÖNEMİ)

Developing prototyping control applications interacting with MathWorks environment removes the algorithm coding task. It is generally not so easy for most students, which requires a special ability, knowledge, and experience. Furthermore, it is naturally a time consuming task [1] for education and industry-oriented applications. On the other hand, it is the fact that the attention and effort should mainly focus on control, implementation and performance and not on coding and debugging [2 and 3].

The main advantage of prototyping control tool, specifically xPC Target used in this study, is the capability to run at much higher sampling rates or to handle more complex control algorithms at a given sampling rate. Also, it runs freely apart from any operating system because of having own special operating system. This is why the xPC Target is preferred in control applications instead of Windows Target.

The real-time prototyping control shortens experiment/project completion time because of suited and fast enough design environment. This approach is an effective tool for preparing the students in control engineering education.

3. REAL-TIME CONTROL IN MATWORKS ENVIRONMENT (MATHWORKS ORTAMINDA GERÇEK ZAMANLI KONTROL)

The heart of rapid control prototyping is automatic code generation [4]. RTW allows rapid prototyping which reduces algorithm coding, designed in blocks with Simulink to an automated process which generates optimized, portable, and customizable ANSI C code from Simulink models [5].

The rapid prototyping also presents some significant advantages over traditional design. The data generated by any component of the model, digital or hardware could be saved in Workspace (MATLAB environment) for later use or displayed on real-time scopes within Simulink. As Simulink model parameters are automatically updated and downloaded to the target hardware while it runs, it presents the opportunity of modifying input or controller parameters and observes the effects of the changes in real time to supervise real-time performance [5].

With RTW, Simulink models can be run in real time on targeted hardware. The RTW includes a set of target files [4] that are compiled by the TLC (Target Language Compiler) to produce ANSI C code. To create a target-specific application, RTW needs a template makefile (*.tmf) so that it can find the appropriate C compiler and compiler options to generate the executable file from the C-coded files [4] for the automatic building process.

The program building process is initiated from Simulink's graphical user interface. After right setup, the real-time algorithm code is automatically generated. The student may configure the building process by modifying the template makefile [4]. Figure 1. shows the rapid prototyping architecture for target applications.

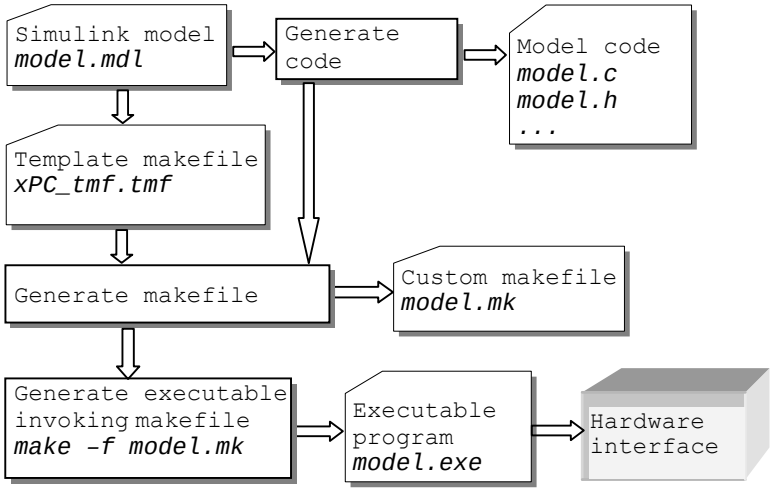

Figure 1. The automatic program building process
(Şekil 1. Otomatik program (kodu) oluşturma işlemi)

For rapid prototyping control applications, The MathWorks offers xPC Target [5], which is a host-target PC solution for prototyping, testing, and deploying real-time systems. It is an environment where the host and target computers are different computers as shown Figure 2.

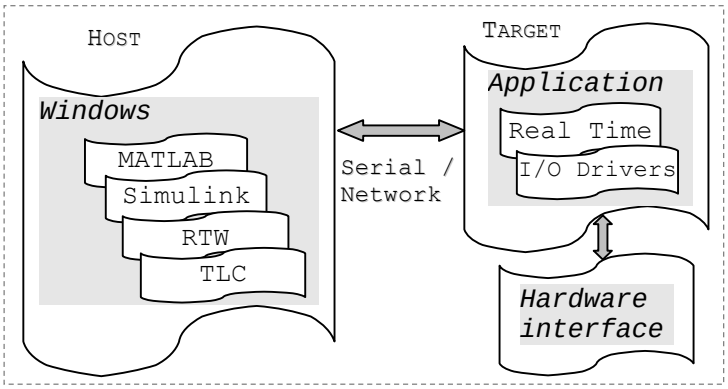

Figure 2. Host-Target real-time control system design
(Şekil 2. Ana Hedef ortamlı gerçek zamanda kontrol sistem tasarımı)

xPC Target is the key in PC-based real-time control environment [5]. It allows adding I/O blocks to the student's Simulink block diagrams to generate code with RTW and downloading it to a second PC compatible hardware that runs the xPC Target real-time kernel.

The xPC Target needs a target PC to execute the real-time code downloaded from the host PC. Installing xPC Target environment needs hardware and software requirements which are widely described in xPC Target toolbox [5].

System configurations for real-time program building are adjusted from Simulink's graphical user interface. The student chooses Simulink in external mode. The remaining adjustment choices are easily completed from simulation parameters dialog box [5].

After the compiling, linking, and downloading the process, a target object is created, the default name is tg. For more information about the target object see the reference [5].

3.1. The Controller Structure (Kontrolör Yapısı)

One of the most useful control algorithms in industry is proportional-integral-derivative (PID) control, which possesses simple, robust, and stable properties [6]. Since the speed control system is linear, a linear controller (PID) is required to get a good control performance.

The analog version of the PID controller is

$$u_a(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \quad (2)$$

where $e(t)$ is the error, $u_a(t)$ is the controller output. K_p, K_i, and K_d are proportional, integral, and derivative constants. The discrete controller of integrator part is constructed as

$$u_i(k) = \frac{T_s}{2}[e(k) + e(k-1)] + u_i(k-1) \quad (3)$$

where $e(k)$ is the error and $u_i(k)$ is integrator output in discrete form and T_s is the sampling time. The differentiator part is

$$u_d(k) = \frac{1}{T_s}[e(k) - e(k-1)] \quad (4)$$

where $u_d(k)$ is differentiator output in discrete form. Finally, the PID controller in discrete form

$$u(k) = K_p e(k) + K_i u_i(k) + K_d u_d(k) \quad (5)$$

The PID controller transfer function in z domain of (5) is obtained in (6) [7].

$$D(z) = K_p + K_i \frac{T_s}{2}\left[\frac{z+1}{z-1}\right] + K_d\left[\frac{z-1}{T_s z}\right] \quad (6)$$

4. EXPERIMENTAL STUDY (DENEYSEL ÇALIŞMA)

Experimental setup shown in Figure 3. mainly consists of two PCs (host and target), two identical PM DC machines, SCR-based DC drive, a small typical PM DC motor (tacho generator), a shunt resistance of 0.25 Ohm, a rheostat of 5 KΩ, and two low pass filters (LPF). The host and target PCs are Pentium II 350 MHz with 128 MB of RAM, running Windows 98. The target PC that is equipped with a data acquisition card, Advantech, PCL-818HG, runs xPC Target real-time kernel.

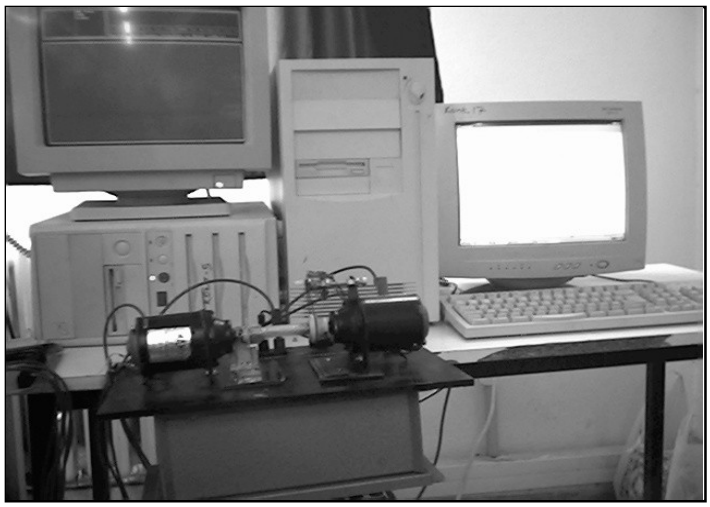

Figure 3. A photo of experimental setup
(Şekil 3. Deney düzeneği fotoğrafı)

One of the PM DC machines coupling to the other's shaft is used as a generator to establish electrical loading. The rheostat is connected to the generator's terminals for loading effect (disturbance). SCR-based DC drive (0-10 V, 5A) drives the PM DC motor (M) coupled to the generator (G). Another small PM DC machine (TG) runs as a tacho-generator to measure the shaft speed of M. The LPFs eliminate the noises from the signals associated with the load current and the shaft speed. The schematic of laboratory setup is depicted in Figure 4.

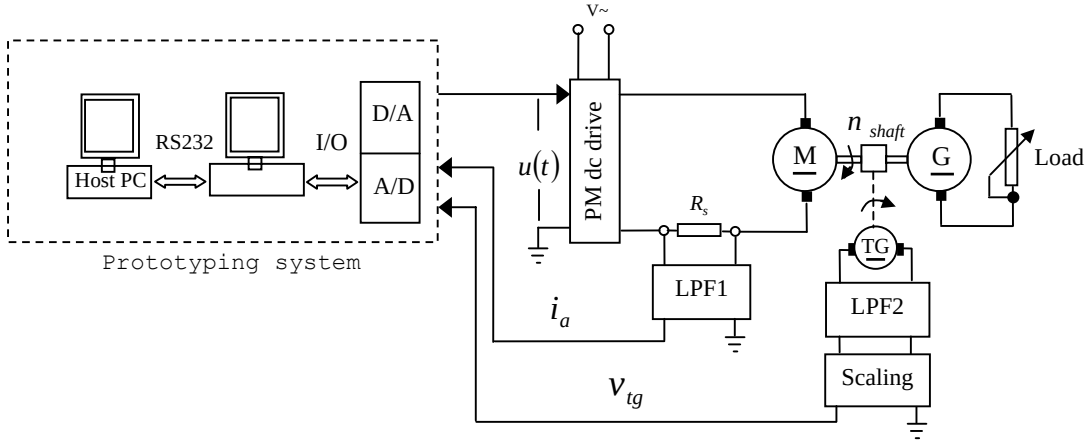

Figure 4. Schematic of laboratory setup
(Şekil 4. Laboratuar düzeneği şeması)

A trimpot standing for an adjustable coefficient K_t is connected to the LPF1's output to obtain a certain ratio between the tacho's shaft speed and the terminal voltage of TG. The block diagram for the TG is illustrated in Figure 5.

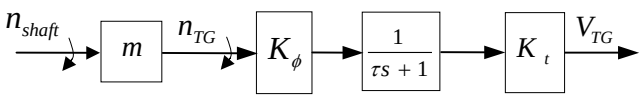

Figure 5. The block diagram of the tacho-generator
(Şekil 5. Tako generatörü blok diyagramı)

The transfer function of TG obtained from Figure 5. is given in equation (7)

$$TG\ (s) = \frac{V_{TG}\ (s)}{n_{shaft}\ (s)} = \frac{mK_{\phi}K_{t}}{\tau s + 1} \qquad (7)$$

where TG(s) indicates the transfer function of TG, n_{shaft} stands for motor shaft speed, which is measured in revolution per minute (rpm), V_{TG} is the TG terminal voltage and m is the ratio of reel diameter on the motor's shaft to the reel diameter on the TG's shaft, which is measured as 5.1. K_t is adjusted to a value of 0.3125 with the trimpot. K_{ϕ} is the generator constant and determined as 0.0032 and LPF2's time constant τ is experimentally determined as 0.047 s.

In rpm calibration measurements, a manual type digital tachometer AMETEK 1726 is used [8]. V_{TG} is adjusted by the trimpot to a value of 2.5V while the digital tachometer reads 2500 rpm. Hence, any n_{shaft} value can be easily found by multiplying V_{TG} by 1000.

In the study, the proper coefficient values for the PID controller are experimentally determined to be K_p=10, K_i=1, and K_d=1.21 for T_s=0.01 s.

4.1. Real-Time Implementation (Geçek Zamanlı Çalışma)

Table 1 gives the necessary steps to develop a real-time implementation with xPC Target. Before creating a Simulink model, the interfacing card driver with the card physically inserted in a target PC slot must be copied to the subdirectory "toolbox/rtw/…/ xPCblock". Subsequently, the student just clicks and drags the I/O block to the Simulink design file.

Table 1. Running the target application in real-time
(Tablo 1. Gerçek zamanlı hedef uygulamasını çalıştırma)

Running procedure
• Create Simulink model.
• Enter Simulation parameters.
• Build and download the target application choosing xPC Target build process.
• Execute the target application in real-time.
• Acquire signals and tune parameters.

Figure 6 shows the implemented Simulink model for the real-time speed control of PM DC motor. The student may change the default settings for the input and output channel number, address, gain, and digital input/output in the Simulink model.

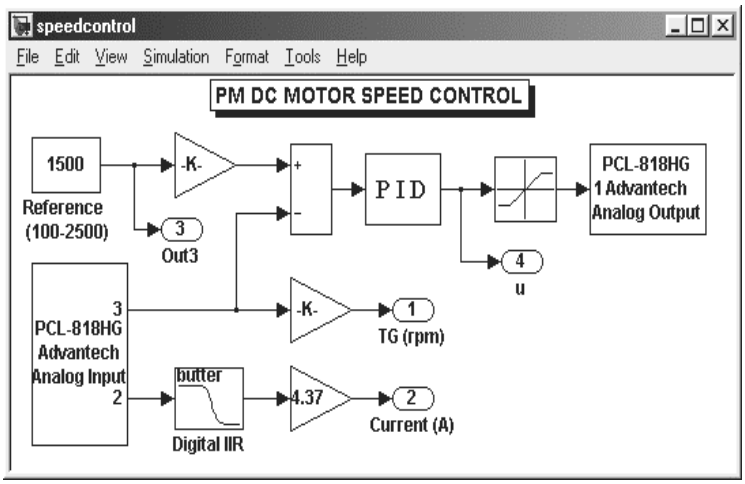

Figure 6. The real-time speed control of PM DC motor with Simulink
(Şekil 6. Simulink ile PM DC motorun gerçek zamanlı hız kontrolü)

Figure 7 shows the flow of information in the system. xPC Target allows the student to change many properties and parameters such as reentering stop time and sample time intervals without rebuilding the target application.

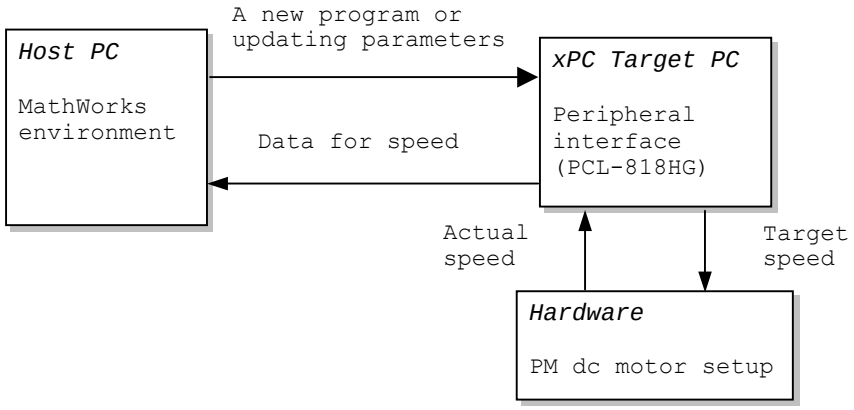

Figure 7. Flow diagram of the speed control system
(Şekil 7. Hız kontrol sistemi akış diyagramı)

Signal logging is the process for acquiring real-time data during a real-time run [5]. After the run reaches its final time or after student manually stops the run, she/he can plot the data, or save it to a disk for later use. For example, after real-time run, typing "data1=tg.OutputLog;" in MATLAB command window, then clicking the open file and right clicking the logged data file (data1), and finally choosing the graph from Workspace, the student can easily get the graphical results. Here, data1 represents a matrix variable that holds the real-time data in matrix form.

Table 2 presents xPC Target commands that are frequently used in real-time implementation in MATLAB command window.

Table 2. Frequently used xPC Target commands
(Tablo 2. Sıkça kullanılan hedef xPC komutları)

Commands and their explanations
tg = xPC; establishes serial connection beetwen host and target PC's.
unload(tg); unloads a target application from the target PC.
load(tg,'motor_pid'); downloads a target application from the host PC to the target PC.
sim('motor_pid'); opens user application designed in Simulink.
tg.StopTime = value; determines run duration of application.
tg.SampleTime = value; enables user to vary the sample time.
tg.ShowParameters = 'on'; tg.px = set value; enables the user to change the block parameters without rebuilding.
+tg or tg.start or start(tg); starts the application.
tg.OutputLog; uploads the logged data to the host PC from the target application.
plot(tg.TimeLog,tg.Outputlog); plot the signals from the outport block.
tg.AvgTET or get(tg,'AvgTET'); gets information about the average task execution time.
-tg or tg.stop or stop(tg); stops the target application.

5. RESULTS AND DISCUSSIONS (SONUÇLAR VE TARTIŞMALAR)

The closed loop (CLC) and open loop control (OLC) are successfully achieved in real time. The results are obtained at different run times and at the same time durations as illustrated in Figures 8, 9, and 10.

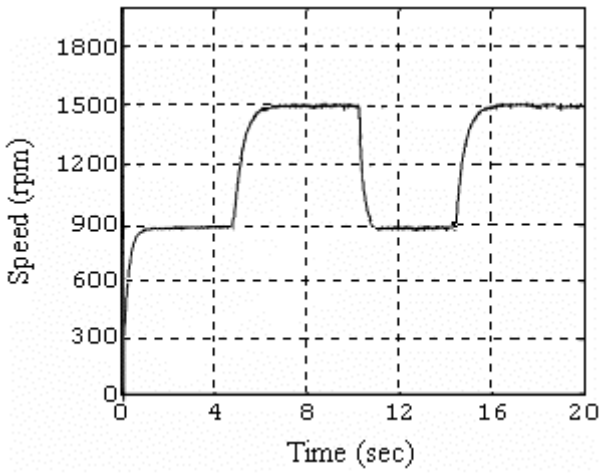

Figure 8. The OLC speed responses to the no load and full load disturbances at the reference of 1500 rpm
(Şekil 8. 1500 rpm referansında yüksüz ve tam yüklü bozucular için açık döngü kontrol hız cevapları)

The disturbances are obtained by abruptly changes from full to no load conditions and vice versa at the constant reference speed of 1500 rpm. In the OLC system shown in Figure 8, as expected, the speed response can not be retained at the reference speed level at the load. The OLC is too sluggish and does not compensate the full load disturbances.

On the other hand, in the CLC system shown in Figure 9 the PID controller successfully compensates the load changes.

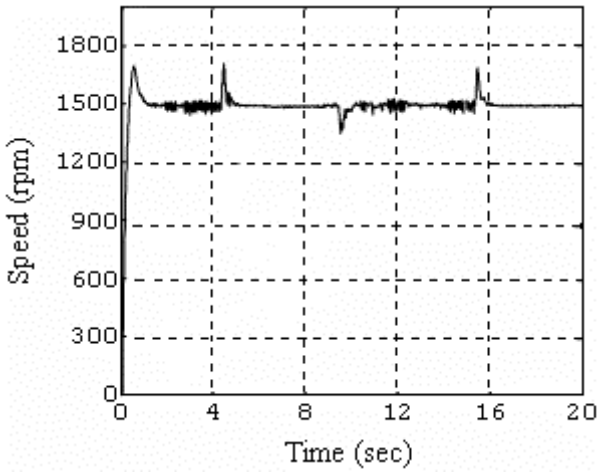

Figure 9. The CLC system speed responses to the no load and full load disturbances at the reference of 1500 rpm
(Şekil 9. 1500 rpm referansında yüksüz ve tam yüklü bozucular için kapalı döngü kontrol sistemi hız cevapları)

Figure 10 gives the graphical results of speed responses to various reference speeds under the full load. At low references such as 600 rpm, the system provides overdamped speed response. There is some overshooting in the output for higher reference speed levels. To make the system behavior even better, the student can try different control strategies in MathWorks environment.

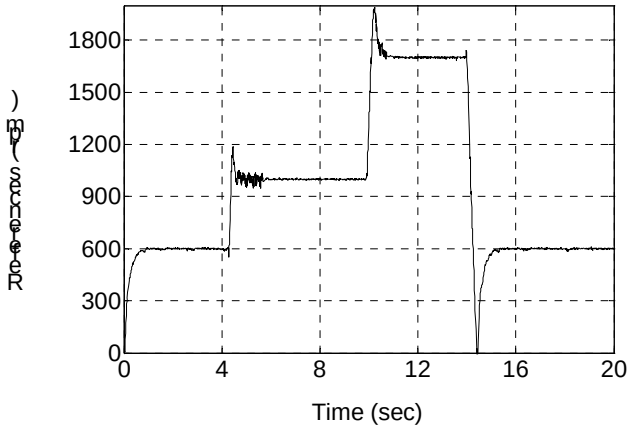

Figure 10. Speed responses of the CLC system to different reference speeds at full load
(Şekil 10. Tam yükte farklı referans hızları için kapalı döngü kontrol sistemi hız cevapları)

6. CONCLUSION (SONUÇ)

A step by step rapid prototyping control guide is provided for control students conducting a real-time implementation in MathWorks environment through a case study. In this platform, students can easily learn how to implement their own control strategies in real time. This sort of rapid prototyping environment increases the quality in learning while drastically reducing hardware costs.

This rapid prototyping control platform has been successfully employed in the Control Laboratory at Marmara University, Electronics and Computer Education Department. This approach is introduced in the "Digital Control Systems I-II" courses taught in both semesters since 2005. On the other hand, Department of Electronic and Computer

Education is a member of an EU supported project, MVET (Modernization of Vocational Education & Training in Turkey). In this project, a modular education program is being adapted to the department curriculum. Because of the successful results, and interests from students, these courses are planned to be included in the curriculum as mandatory courses.

Since the students can try their algorithms, obtain and evaluate their results, and show them graphically in a typical two-hour lab session, it is clearly observed that this kind of "hands on" approach increases students' interest and reinforce learning.

REFERENCES (KAYNAKLAR)

1. Teng, F.C., (2000). Real-Time Control Using MATLAB Simulink, IEEE, 4, pp:2697.
2. Burns, D. and Sugar, T.G., (2002). Rapid Embedded Programming in the Mathworks Environment, ASME, 2, pp:237.
3. Shiakolas, P.S. and Piyabongkarn, (2001). On the Development of a Real-Time Digital Control System Using xPC- Target and a Magnetic Levitation Device, Conference on Decision and Control, Proceedings of the 40th IEEE), pp:1348-1353.
4. Gan, W.S., Chong, Y.K., Gong, W., and Tan, W.T., (2000). Rapid Prototyping System for Teaching Real-Time Digital Signal Processing, IEEE, Transactions On Education, 43, pp:19-21.
5. The MathWorks Inc, (2004). xPC Target For Use with Real-Time Workshop.
6. Shieh M.Y. and Li, T.H.S., (1995). Implementation of Integrated Fuzzy Logic Controller for Servomotor System, IEEE, Proceedings of Fuzzy Systems Int. Conf., 4, pp:1756.
7. Onat, M. and Doğruel, M., (2004). Fuzzy Plus Integral Control of the Effluent Turbidity in Direct Filtration, IEEE Transaction on Control Systems Technology', 12, pp:71.
8. www.ametek.com/products, (2006), AMETEK Corporate Office 37 N. Valley Road, Building 4 P.O. Box 1764 Paoli, PA 19301 USA)