Hardware Implementation of a Neural Network Controller with an MCU and an FPGA for Nonlinear Systems

Sung-Su Kim and Seul Jung*

Abstract: This paper presents the hardware implementation of a neural network controller for a nonlinear system with a micro-controller unit (MCU) and a field programmable gate array (FPGA) chip. As an on-line learning algorithm of a neural network, the reference compensation technique has been implemented on an MCU, while PID controllers with other functions such as counters and PWM generators are implemented on an FPGA chip. Interface between an MCU and a field programmable gate array (FPGA) chip has been developed to complete hardware implementation of a neural controller. The developed neural control hardware has been tested for balancing the inverted pendulum while controlling a desired trajectory of a cart as a nonlinear system.

Keywords: FPGA, MCU, neural controller, reference compensation technique, VHDL.

1. INTRODUCTION

PID controllers are still predominantly used in the industry for most motion control applications, because of their simplicity, real-time control capability, easy implementation, and cost effectiveness. PID controllers, however, may not work properly if system parameters change or outer disturbances are present. In addition, PID controllers for nonlinear systems do not work as expected due to the fact that fixed-controller gains have lack of flexibility to deal with nonlinear effects.

To remedy this defect of PID controllers, many control algorithms have been presented. The adaptive control method has been used to tune PID controller gains against system parameter variations. The robust control method has been used to reject disturbances. Advanced control theories have been well developed to tackle system uncertainties, and although they may solve problems associated with the PID controller, they require system dynamic models to derive the adaptive control laws and the robust control laws. An added challenge is that the required system dynamic models are often unavailable or hard to obtain.

Recently, intelligent control methods have been widely used and are gradually being accepted in motion control industries as well as control system communities. A merit of intelligent control methods is that they do not require a system model. They gradually adapt and learn if system parameter variations and disturbances exist. Intelligent tools, such as neural network, genetic algorithms, or fuzzy logic, can be applied on their own or by fusing two or more tools to improve overall performance. Many successful control applications to the nonlinear system such as an inverted pendulum system, have been presented in the literature [1-7].

For a nonlinear system, a neural network is a good choice as a nonlinear controller, and works quite well by compensating for unknown uncertainties [8]. Successful neural network applications can be found in such control systems as, controlling robot manipulators, as a highly nonlinear, multi-input, multi-output (MIMO) system [9,10], and inverted pendulum systems, as a single-input, multi-output (SIMO) system [11,12].

Although the neural network can be used quite successfully for nonlinear systems, problems can occur due to a real-time implementation issue of an extensive-computing requirement of neural network learning algorithm. With the help of the DSP hardware technologies, successful real-time neural network applications have become feasible. A typical neural network application example has been conducted to control a two degrees-of-freedom inverted pendulum system [12].

To make the cost-effective intelligent controller, we have developed intelligent control hardware by combining an embedded controller on a Field Programmable Gate Array (FPGA) and a low cost DSP board for general purpose. The DSP board is,

Manuscript received June 28, 2005; revised May 8, 2006; accepted May 15, 2006. Recommended by Editor Jin Young Choi. This work was supported by the Korea Science and Engineering Foundation under grant KOSEF 2003-000-10389-0.

Sung-Su Kim and Seul Jung are with BK21 Mechatronics Group, Chungnam National University, 220 Gung-dong, Yuseong-gu, Daejeon 305-764, Korea (e-mails: a74110328@hanmail.net, jungs@cnu.ac.kr).

* Corresponding author.

however, expensive compared to microprocessors, and have the speed and computing power required to handle neural network learning algorithm [13].

In this article, we implement a neural network control algorithm in a microcontroller unit (MCU) to achieve cost-effectiveness. An FPGA chip is designed for embedded-PID controllers, by using a very high speed integrated circuit hardware description language (VHDL). Combining an FPGA chip and an MCU forms the low-cost intelligent neural network controller. Recently, in the control system industry, the concept of a controller-on-chip has been expanded, and FGPA chips have been widely used in many control applications [14-16].

Using the high capacity of an FPGA chip, the additional hardware such as an encoder counter and a pulse-width-modulation (PWM) generator, can also be embedded into a single FPGA chip. As a result, cost effectiveness and space-saving benefits can be achieved.

To show the performance of the developed intelligent controller, an inverted pendulum system is tested. The controller is required to control the balance of the pendulum and the position tracking of the cart simultaneously.

2. OVERALL SYSTEM STRUCTURE

The overall control block diagram is shown in Fig. 1. A neural network is placed in front of the closed loop controlled system, functioning as a pre-filter to modify reference trajectories [9]. The system is controlled by a main controller such as a PD, PI, or PID controller. Neural network is added as an auxiliary controller to compensate for uncertainties in the system. This method is known as the reference compensation technique (RCT) which is one of the on-line learning control methods in neural network control applications. One typical structural merit of this control method is that the neural compensating block can be separated physically from the controlled system, and connected by wireless communication.

Fig. 1 describes how the compensating signal from neural network is added to the output error.

$$e = r - y + \phi, \tag{1}$$

where ϕ is a neural network output.

Then the compensated error e is multiplied by a

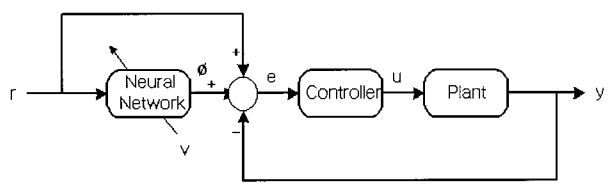


Fig. 1. Reference compensation control structure.

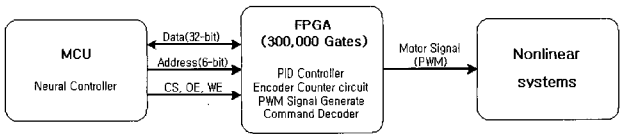


Fig. 2. Overall control structure block diagram.

feedback controller gain. The learning signal v is designed to make the output error converge to zero. Selecting the learning signal is the key issue in on-line learning control applications, and leads to the definition of different control structures [9,10]. One of the simplest ways of selecting the learning signal is the output error.

$$v = r - y. \tag{2}$$

The PID controller is used as a main controller, and embedded into an FPGA chip. The neural network control algorithm is embedded on the MCU. Fig. 2 shows the hardware block diagram structure of the interfacing between each module of the intelligent controller. The MCU board communicates with the FPGA to synchronize the data flow. The MCU gives compensated signals to the FPGA chip, and the FPGA adds those compensated signals to output errors. PID controllers generate control input signals, which are then converted into PWM signals to drive DC motors.

3. EMBEDDED PID CONTROLLER

3.1. Overall structure

The embedded PID controller on an FPGA chip consists of a communication block, an encoder counter block, a PID calculation block, and a PWM generation block. Fig. 3 shows the block diagram inside the PID controller. Input signals to the PID controller are a 32 bit data bus, a 6 bit address bus, control signals such as CS, OE, WE, encoder signals, and a 25MHz clock. Output signals are PWM signals. Encoder measurement signals are counted and compared with desired values, and the errors are formed. Errors are used in PID control calculation, and PID controller output generates PWM signals to the motor drivers. The PID controllers are imple-

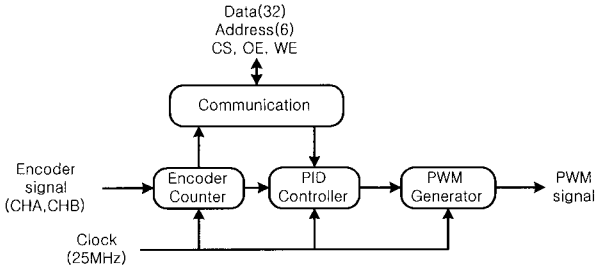


Fig. 3. Block diagram of an embedded PID controller.

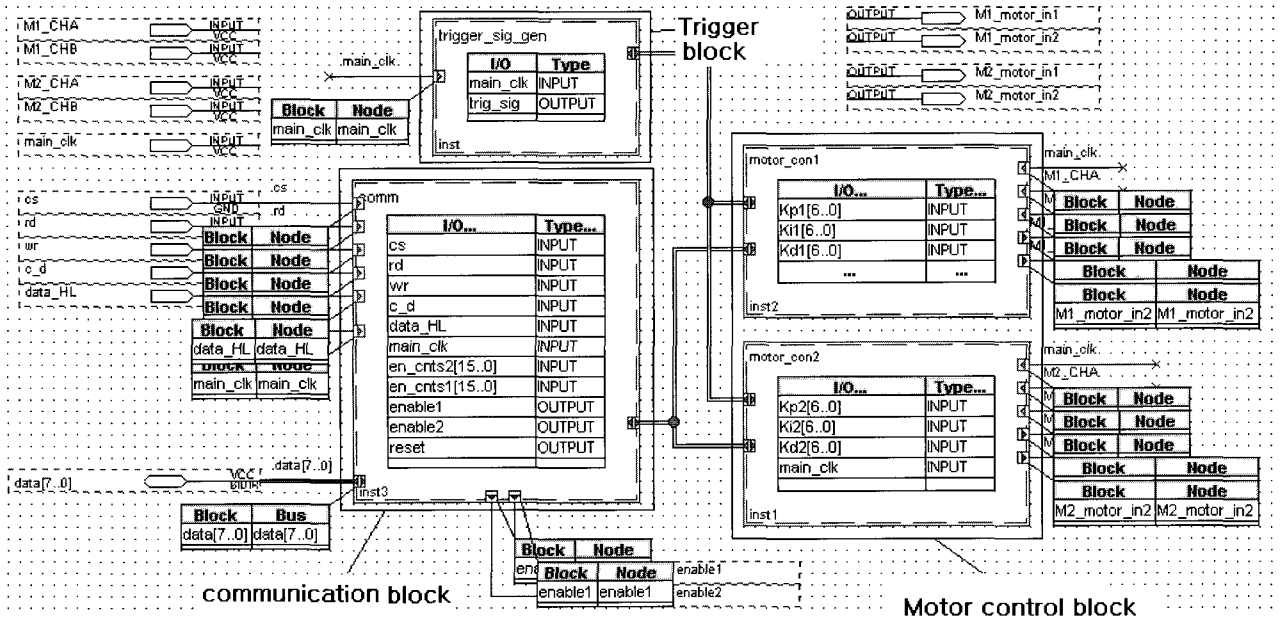


Fig. 4. Schematic design of PID controller.

mented in an FPGA chip containing 300,000 gates. The actual schematic design of the controller is shown in Fig. 4. PID motion controller design is programmed by Quartus II 2.0.

3.2. Communication block

The communication block receives PID gains and desired trajectory from the MCU, and then transfers encoded data from the controller to an MCU whenever it is needed. Data write, from an MCU to the motion controller, includes PID gains, enable/disable signals and the motion controller reset. Data read, from the motion controller to an MCU, is to read encoder data. Since the size of the encoder data is 16 bit, high and low bytes should be read separately.

3.3. Motor control block

Fig. 5 shows the motor control block. It consists of an encoder counter block, a PID controller block, and a PWM generator block. The clock synchronizes the process between each module. The trigger signal is generated at each 1 kHz sampling time. The trigger signal is sent to the encoder counter block and the PID controller block and synchronizes the process of each block. The block takes PID gains, a clock, a trigger signal, encoder signals and a reset signal as input, and PWM signals and encoder counter values as output.

3.4. Encoder counter block

The counter block counts and determines the direction of motor rotation from encoder signals. The block diagram is shown in Fig. 6. Differences in phase A and phase B determine the direction of rotation. Noise from a mechanical system can be filtered out by

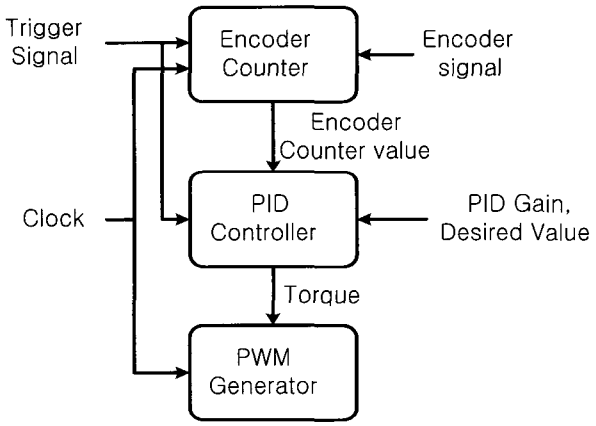


Fig. 5. Structure of motor control block.

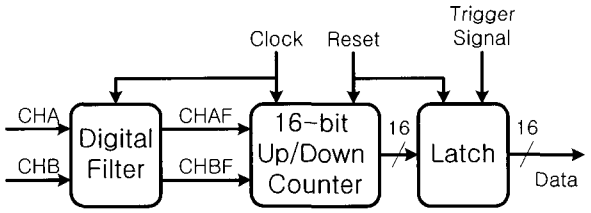


Fig. 6. Structure of encoder counter block.

a digital filter. Every trigger signal is able to generate counter values. 16 bits of the counter, limits the range of the movements.

3.5. PID controller block

The controller block receives encoder data from the encoder counter block. It then compares the data with the desired values to generate positional errors and generate PID control torque. PID control equations are

$$\tau(n) = K_p e(n) + K_i s(n) + K_d \{e(n) - e(n-1)\}, \quad (3)$$

where $\tau(n)$ is control input, $e(n)$ is error, K_p, K_i, K_d are PID gains, and

$$s(n) = \begin{cases} s_i & \sum e(n) > s_i \\ \sum e(n) & -s_i \le \sum e(n) \le s_i \\ -s_i & \sum e(n) < -s_i, \end{cases} \quad (4)$$

where s_i is a threshold value.

4. NEURAL NETWORK CONTROL ALGORITHM

4.1. Reference compensation technique

In this paper, we are presenting an on-line learning algorithm for neural network. The idea of the RCT is that the neural network compensates at the input level, to modify input signals for minimizing output errors. The same objective function of the feedback error learning method is minimized in on-line fashion [9,10].

The angle error of the pendulum is formed as

$$e_\theta = \theta_d - \theta, \quad (5)$$

where θ_d is a desired angle, and θ is an actual angle.

A PID controller for an angle control is defined as

$$u\theta = k_{P\theta} e_\theta(t) + k_{i\theta} \int e_\theta(t)dt + k_{d\theta} \dot{e}_\theta(t) + k_{P\theta}\phi_1 + k_{d\theta}\phi_2 + k_{i\theta}\phi_3, \quad (6)$$

where ϕ_1, ϕ_2, ϕ_3 are neural network outputs. The cart position is controlled, as well as the pendulum angle. The position tracking error is formed as

$$e_x = x_d - x, \quad (7)$$

where x_d, x are the desired cart position and actual cart position, respectively. A PID controller for a cart can be formed as

$$ux = k_{Px} ex(t) + k_{ix} \int ex(t)dt + k_{dx} \dot{ex}(t) + k_{ix}\phi_4 + k_{dx}\phi_5 + k_{px}\phi_6, \quad (8)$$

where ϕ_4, ϕ_5, ϕ_6 are neural network outputs. The overall control input for the inverted pendulum system is

$$u = u_x + u_\theta. \quad (9)$$

Fig. 7 shows the control block diagram for an inverted pendulum system.

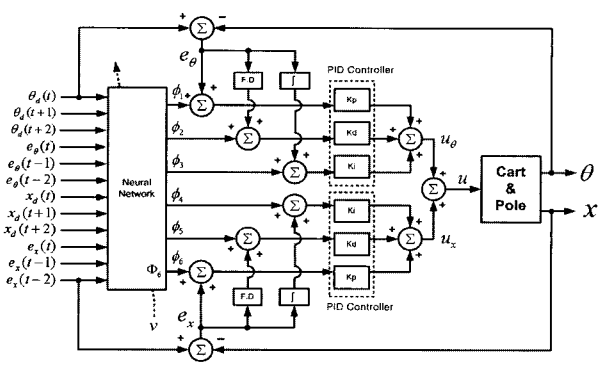


Fig. 7. Neural network control for an inverted pendulum system.

4.2. Neural network learning algorithm

For a neural network structure, we have used a general feed-forward structure that has an input layer, a hidden layer, and an output layer. Inputs to neural network are the delayed values of an angle error and a position error with the desired values of the angle and the position.

The numbers of an input layer neuron, a hidden layer neuron, and an output layer neuron are 12, 9, and 6 respectively as shown in Fig. 8. For a nonlinear function at a hidden layer and an output layer, we have used a hyperbolic tangent function, shown as

$$f(x) = \frac{1 - \exp(-x)}{1 + \exp(-x)}. \quad (10)$$

Here, the neural network learning algorithm is derived Since we are doing on-line learning and control, the selection of the learning signal is quite important. Minimizing the training signal makes the output error converge to zero. Neural network outputs are defined as a sum of compensating signals.

$$\Phi = \Phi_\theta + \Phi_x, \quad (11)$$

where

$$\Phi_\theta = k_{p\theta}\phi_1 + k_{d\theta}\phi_2 + k_{i\theta}\phi_3,$$

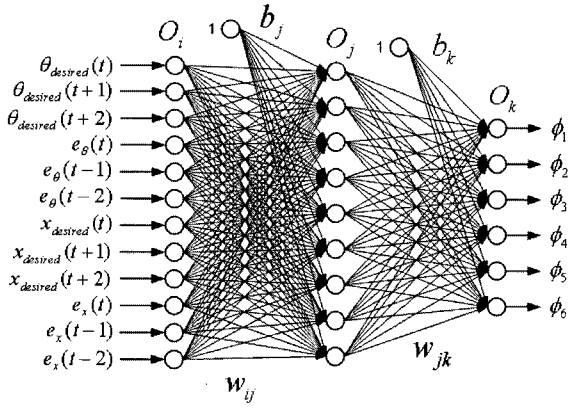


Fig. 8. Neural network structure.

$$\Phi_x = k_{px}\phi_4 + k_{dx}\phi_5 + k_{ix}\phi_6.$$

If $f(\theta, \dot{\theta}, \ddot{\theta})$ is assumed to be a system dynamics, (11) can be represented below.

$$k_{p\theta}e_\theta + k_{d\theta}\dot{e}_\theta + k_{i\theta}\int e_\theta + k_{px}e_x + k_{dx}\dot{e}_x + k_{ix}\int e_x = f(\ddot{\theta},\dot{\theta},\theta,\ddot{x},\dot{x},x) - \Phi. \quad (12)$$

If the left side of (12) becomes zero, then $\Phi \cong f(\theta, \dot{\theta}, \ddot{\theta})$. This means that inverse dynamics control of the dynamical system can be achieved.
So here, we define the learning signal of neural network as the PID controller output.

$$v = k_{p\theta}e_\theta + k_{d\theta}\dot{e}_\theta + k_{i\theta}\int e_\theta dt + k_{px}e_x + k_{dx}\dot{e}_x + k_{ix}\int e_x dt. \quad (13)$$

The objective function is defined as a positive constant.

$$E = \frac{1}{2}v^2. \quad (14)$$

Differentiating (14) with respect to the neural network weights, we have

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial v}\frac{\partial v}{\partial w} = v\frac{\partial v}{\partial w} = -v\frac{\partial \Phi}{\partial w}. \quad (15)$$

Here, we have

$$\frac{\partial E}{\partial w} = -v\frac{\partial \Phi}{\partial w} = -v\left(\frac{\partial \Phi_\theta}{\partial w} + \frac{\partial \Phi_x}{\partial w}\right). \quad (16)$$

The update equation in the back-propagation algorithm [17] is

$$\Delta w(t) = \eta\frac{\partial \Phi}{\partial w}v + \alpha\Delta w(t-1), \quad (17)$$

$$w(t+1) = w(t) + \Delta w(t), \quad (18)$$

where η is the learning rate and α is the momentum coefficient.

4.3. Hardware implementation of a neural network controller

A commercially available advanced RISC micro-processor board (ARM), manufactured by Samsung is used for a neural network controller implementation. The ARM board has a 32 bit, 66 MHz RISC structured microprocessor. In order for an ARM board to communicate with embedded PID controllers on the FPGA, a 32 bit data bus is used to share data.

At every sampling time, the ARM board must give compensating values to the FPGA, so that the FPGA can add those values to output errors to form PID controllers. This kind of process can be done, within one sample time, by a communication module between the MCU and the FPGA.

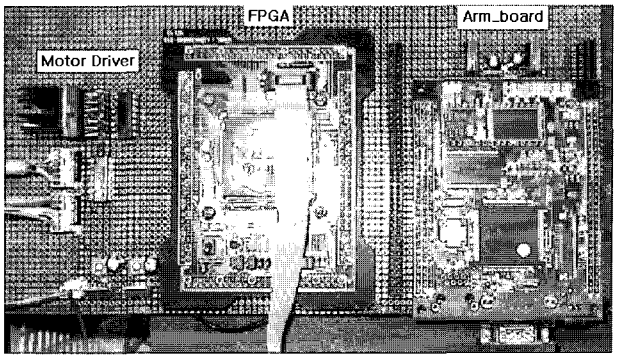


Fig. 9. MCU-FPGA intelligent control hardware.

Fig. 9 shows a real figure of the embedded controller, consisting of an MCU, an FPGA, and a motor driver.

5. EXPERIMENTS

5.1. Experimental setup

Experiments are conducted to control the angle of pendulum and the position of cart, simultaneously. Fig. 10 shows the experimental setup of the inverted pendulum system. The system consists of an inverted pendulum, an embedded hardware controller, and a PC.

5.2. Pendulum balancing control

We have found that the maximum sampling time for on-line learning and control can be achieved at 10 ms. This sampling rate is much slower than that of a DSP board, and although the ARM board is fast enough, computing back-propagation learning algorithm can be a burden.

For control applications, however, 10ms sampling time is acceptable. First, PID controllers are tested to balance the pendulum without moving the cart. As expected [13], PID controllers can balance the pendulum while the cart position continues to move, which means that position of the cart is not controlled.

Next experiment is to add a neural network

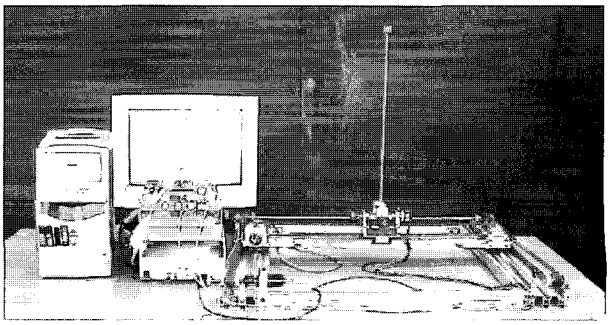


Fig. 10. Inverted pendulum system.

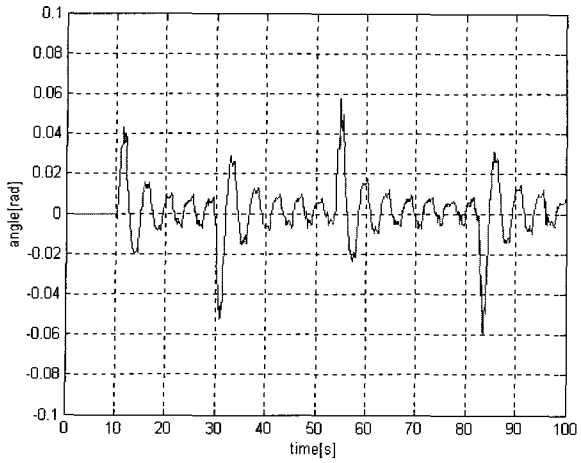


Fig. 11. Angle of the pendulum.

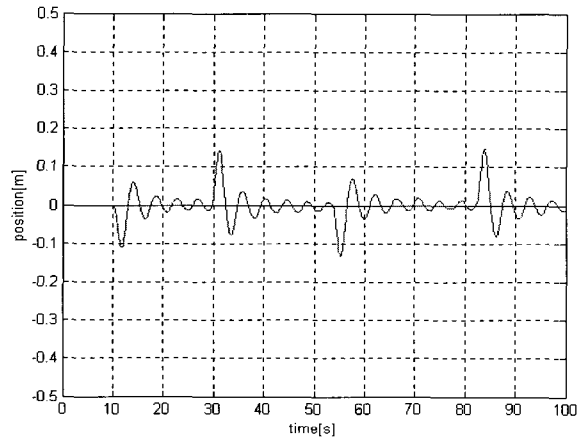


Fig. 12. Position of the cart.

controller. Figs. 11 and 12 show the pendulum angle error and cart position error controlled by the controller, respectively. The pendulum is well maintained at balance. Peaks occur if disturbances are present by intentionally hitting the pendulum.

5.3. Cart position control

The cart is required to track a desired sinusoidal trajectory while balancing the pendulum. The cart is required to move along x axis a distance of approximately 40cm. PID controllers are tested, but fail.

To improve the tracking performance of the cart, the estimation of velocity should be filtered. Here we tested two cases, one is with a digital filter being used, and the other is not. The filter is used to smooth the approximated velocity value obtained from the numerical calculation.

1) Case 1: Without a digital filter

We have found that derivative terms in the PID controller, after approximation by a finite difference computation, are quite noisy. Although simultaneous pendulum balancing and position tracking control

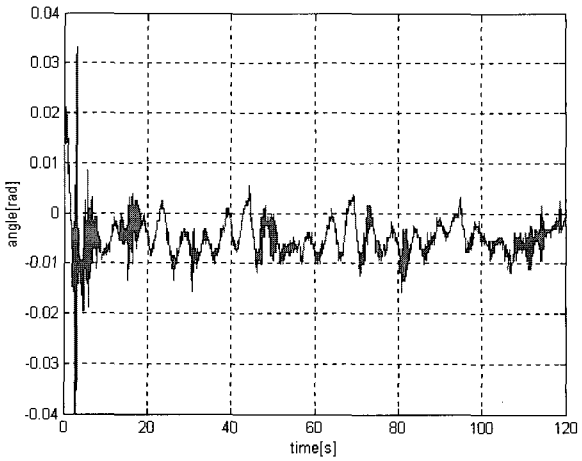


Fig. 13. Case 1: Pendulum angle of a sinusoidal trajectory tracking task when T = 8π sec .

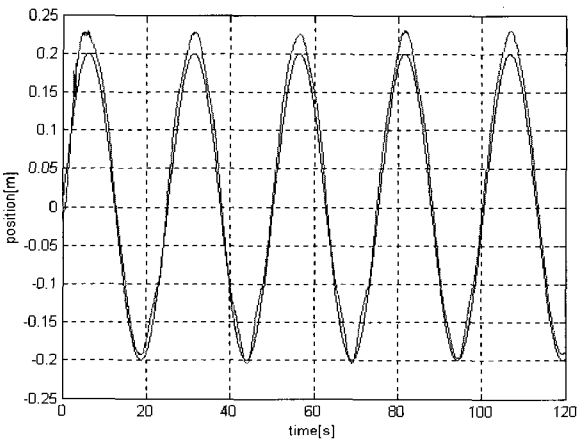


Fig. 14. Case 1: Position tracking of the cart of a sinusoidal trajectory tracking task, when T = 8π sec .

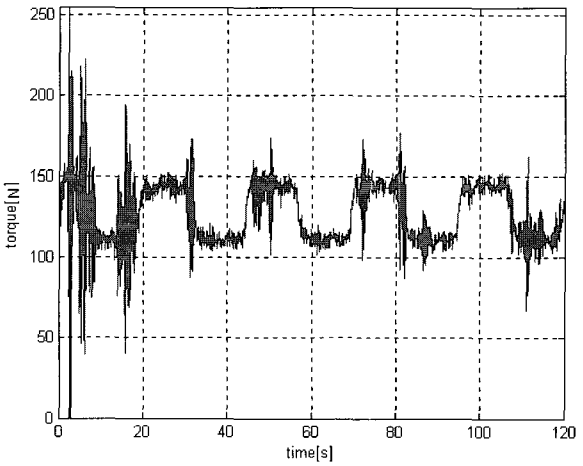


Fig. 15. Case 1: Control torque of a sinusoidal trajectory tracking task, when T = 8π sec .

tasks are successful, vibration is observed. Fig. 13 shows the performance of the balancing pendulum. The pendulum is well maintained, and the angle error

is less than 0.01 radian.

Fig. 14 shows the tracking performance of the cart while balancing the pendulum. The period of the sinusoidal trajectory is about 25 seconds. We observed that an overshoot of about 3 cm occurred in cart position tracking. To minimize the overshoot, the smoothing filter is used. Fig. 15 shows the corresponding torque, with glitches causing vibration. To eliminate vibration, we used a digital filter.

2) Case 2: With a digital filter

We use a digital filter for smoothing the signal after a finite difference process. We have experimentally found the vibrating frequency at around 5Hz. So we filter derivative terms out with a low pass filter at the cutoff frequency of 5 Hz. The 2nd order IIR filter is designed,

$$H(z)=\frac{0.75596+0.511922z^{-1}+0.7559611z^{-2}}{1+0.45445z^{-1}+0.572398z^{-2}}.$$

The results are shown in Fig. 16. We can see that

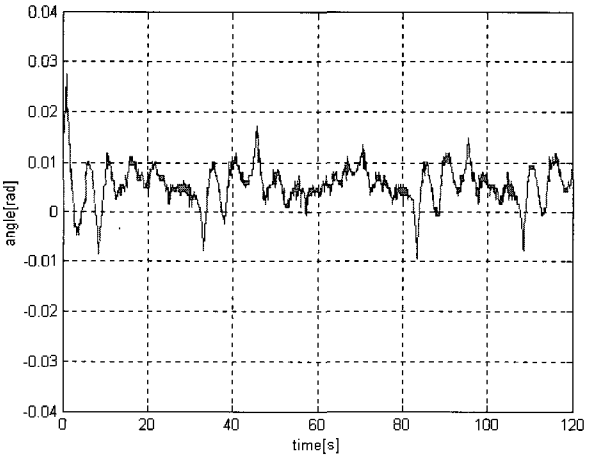


Fig. 16. Case 2: Pendulum angle of a sinusoidal trajectory tracking task with a filter.

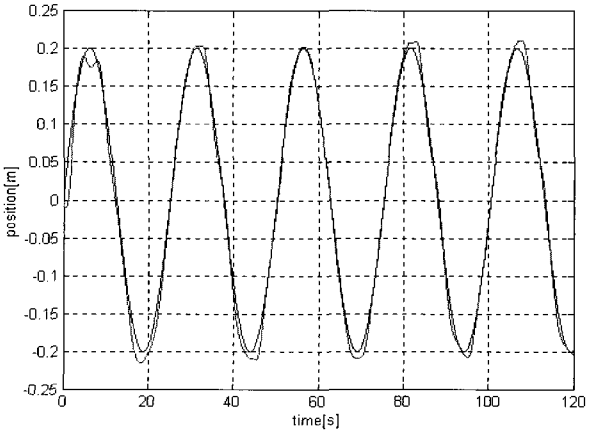


Fig. 17. Case 2: Position tracking of the cart of a sinusoidal trajectory tracking task, when T = 8π sec.

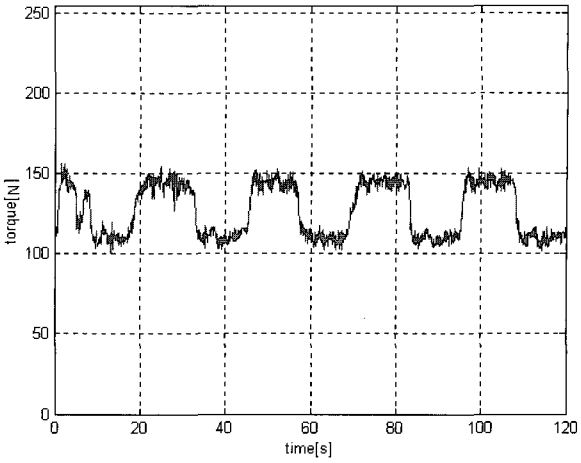


Fig. 18. Case 2: Control torque of the cart of a sinusoidal trajectory tracking task.

the performance results are much better than those without a filter. Specifically, for the cart position tracking, overshoots have been minimized, as shown in Fig. 17. We can clearly see from Fig. 18 that torque is much smoother than that of case 1. In these experiments, the pendulum is balanced within a 0.015 radian angle error, and the cart tracks desired trajectories within the error of 1 cm.

6. CONCLUSION

This paper presents the hardware implementation of a neural network controller, by combining an embedded-PID controller and an MCU board. The controller was tested by performing control of an inverted pendulum system. Although the overall sampling time is slower than that of a DSP, the controller successfully balances the pendulum, while controlling the cart position tracking. The pendulum is balanced within a 0.015 radian angle error, and the cart tracks desired trajectories within the error of 1 cm.

The motivation for combining an MCU and a low level FPGA, was cost effectiveness. Implementing neural network on a single FPGA is very difficult, and requires expensive hardware for the larger size FPGA chips required for on-line back-propagation learning algorithm.

For future research, a PID controller and neural network can be embedded in a single FPGA chip.

REFERENCES

[1] C. W. Anderson, "Learning to control an inverted pendulum using neural networks," *IEEE Control Systems Magazine*, vol. 9, no. 3, pp. 31-37, April 1989.

[2] I. Fantoni and R. Lozano, "Global stabilization of the cart-pendulum system using saturation functions," *Proc. of IEEE Conference on Decision and Control*, pp. 4393-4398, 2003.

[3] R. Yang, Y. Y. Kuen, and Z. Li, "Stabilization of a 2-DOF spherical pendulum on x-y table," *Proc. of IEEE Conference on Control Applications*, pp. 724-729, 2000.

[4] R. J. Wai, J. D. Lee, and L. J. Chang, "Development of adaptive sliding mode control for nonlinear dual-axis inverted-pendulum system," *Proc. of IEEE/ASME Conference on Advanced Intelligent Mechatronics*, pp. 815-820, 2003.

[5] T. Lahdhiri, C. Carnal, and A. Alouani, "Cart-pendulum balancing problem using fuzzy logic control," *Proc. of Southeastern Conference*, pp. 393-397, 1994.

[6] M. E. Magana and F. Holzapfel, "Fuzzy-logic control of an inverted pendulum with vision feedback," *IEEE Trans. on Education*, vol. 41, no. 2, pp. 165-170, 1998.

[7] T. H. Hung, M. F. Yeh, and H. C. Lu, "A pi-like fuzzy controller implementation for the inverted pendulum system," *Proc. of IEEE Conference on Intelligent Processing Systems*, pp. 218-222, 1997.

[8] W. T. Miller, R. S. Sutton, and P. J. Werbos, *Neural Networks for Control*, The MIT Press, 1991.

[9] S. Jung and T. C. Hsia, "Neural network inverse control techniques for PD controlled robot manipulator," *Robotica*, vol. 19, no. 3, pp. 305-314, 2000.

[10] H. Miyamoto, K. Kawato, T. Setoyama, and R. Suzuki, "Feedback error learning neural network for trajectory control for of a robotic manipulator," *Neural Networks*, vol. 1, pp. 251-265, 1988

[11] S. Jung and H. T. Cho, "Decentralized neural network reference compensation technique for PD controlled two degrees-of-freedom inverted pendulum," *International Journal of Control, Automations, and System*, vol. 2, no. 1, pp. 92-99, 2004.

[12] S. Omatu, T. Fujinaka, and M. Yoshioka, "Neuro-pid control for inverted single and double pendulums," *Proc. of IEEE Conf. on Systems, Man, and Cybernetics*, pp. 8-11, 2000.

[13] S.-S. Kim and S. Jung, "Hardware implementation of real time neural network controller with a DSP and an FPGA for nonlinear systems," *Proc. of IEEE Conf. on Robotics and Automations*, pp. 4639-4644. 2004.

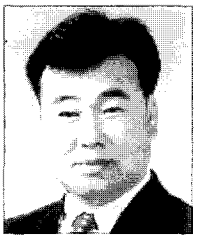
[14] M. Krips, T. Lammert, and A. Kummert, "FPGA implementation of a neural network for a real-time hand tracking system," *Proc. of the First IEEE International Workshop on Electronic Design, Test and Applications*, pp. 313-317, 2002.

[15] M. Cristea, J. Khor, and M. McCormick, "FPGA fuzzy logic controller for variable speed generators," *Proc. of the IEEE International Conference on Control Applications*, pp. 301-304, 2001.

[16] A. Kongmunvattana and P. Chongstivatana, "A FPGA-based behavioural control system for a mobile robot," *Proc. of the IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 759-762, November 1998.

[17] J. L. McClelland and D. E. Rumelhart, "Explorations in parallel distributed processing," *The MIT Press*, 1991.

Sung-su Kim received the B.S. degree in Control and Measurement Engineering from The Kyung-Il University in 2001 and the M.S. degree in Mechatronics Engineering from Chungnam National University in 2004. He is presently at the Korea Atomic Energy Research Institute. His interests include S.O.C design, DSP applications, and robotics.



Seul Jung received the B.S. degree in Electrical & Computer Engineering from Wayne State University in 1988, and the M.S. and Ph.D. degrees in Electrical & Computer Engineering from the University of California, Davis in 1991 and 1996, respectively. After working at the Advanced Highway Maintenance and Construction Technology Center, he joined the Department of Mechatronics Engineering, Chungnam National University in 1997, where he is presently an Associate Professor. His research interests include intelligent systems, hardware implementation of intelligent controllers and intelligent robotic systems. He is a Member of Tau Beta Pi and Eta Kappa Nu.