Implementation of an Intelligent Controller with a DSP and an FPGA for Nonlinear Systems

Sung-Su Kim* and Seul Jung**

Intelligent Systems and Emotional Engineering Lab.
Department of Mechatronics Engineering
Chungnam National University, Daejeon, Korea
(+82-42-821-6876, *a74110328@hanmail.net, **jungs@cnu.ac.kr)

Abstract: : In this paper, we develop a control hardware such as an FPGA based general purpose controller with a DSP board to solve nonlinear control problems. PID control algorithms are implemented in an FPGA and neural network control algorithms are implemented in a DSP board. PID controllers implemented on an FPGA was designed by using VHDL to achieve high performance and flexibility. By using high capacity of an FPGA, the additional hardware such as an encoder counter and a PWM generator, can be implemented in a single FPGA device. As a result, the noise and power dissipation problems can be minimized and the cost effectiveness can be achieved. In order to show the performance of the developed controller, it was tested for controlling nonlinear systems such as an inverted pendulum.

Keywords: FPGA, VHDL, DSP, Intelligent controller

1. INTRODUCTION

The PID controller is a well known and easily implemented device for motion control applications. It works very well for linear systems with optimal gain tunings. However, for the nonlinear systems, plant dynamic is varying so that fixed PID controller gains do not work properly as they should. Due to system parameter variations and external disturbances, the performance of the PID controller is degraded. This actually leads to the need of nonlinear controllers for nonlinear systems.

As a nonlinear controller, neural network is a one good candidate that works very well for nonlinear systems by compensating for unknown uncertainties[1]. Applications of neural network can be found in many areas such as motion control system, signal processing, and data processing. Successful neural network application can be found in controlling robot manipulators since they are highly nonlinear MIMO systems[2-3].

Even though neural network works well, the problem comes when real time implementation of fast computing time of neural network algorithm is required. Simulation studies of neural network control can be easily done and seen in the literature. Experiments are more difficult to be conducted since fast computing devices are required. The control-loop in the neural control system must be processed at every sampling period. Recently, with the help of high-performance and high speed hardware technology, micro-controllers such as DSPs and MCUs are available for fast computation in the market. However, commercial DSP board packages such as dSpace and the Math Works are very expensive in order to have full capacity.

In our previous researches, successful real time neural network applications have been achieved in controlling an x-y table robot [4]. Another typical neural network application example was conducted to control an inverted pendulum [4,5]. Those works have been done by using a commercial DSP board package.

In this paper, as an extension our previous researches, we develop a intelligent control hardware such as an FPGA (Field Programmable Gate Array) based general purpose controller with a DSP board to deal with nonlinear problems. Actually, this paper is the extended part of our paper that deals with development of PID controllers on an FPGA chip [6].

We extend our previous PID controller implemented in an FPGA to have neural network control algorithms in a DSP board as well. An FPGA was designed by using VHDL to achieve high performance and flexibility. Recently, by the concept of controller on chip, FGPA chips have been used in many applications [7-12].

By using the high capacity of an FPGA, the additional hardware such as an encoder counter and a PWM generator, can be implemented in a single FPGA device. As a result, the noise and power dissipation problems can be minimized and it shows the cost effectiveness. A commercially available general purpose low cost DSP board is used to implement neural network controller. Back propagation learning algorithm has been implemented. Interface between a DSP board and an FPGA has been done to form an intelligent control hardware.

In order to show the performance of the developed controller, it was tested for controlling nonlinear systems such as an inverted pendulum. The proposed controller is required to control the balance of the pendulum and the position tracking of the cart simultaneously. Performances of position tracking of the cart while balancing the pendulum were successfully achieved.

2. OVERALL SYSTEM STRUCTURE

The reference compensation technique is known as one of on line learning control methods of neural network application. The control block diagram is shown in Fig. 1. Neural network placed in front of the closed loop control system functions as a pre-filter to modify reference trajectories [2].

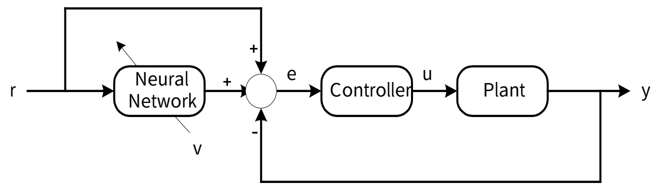

Fig.1 Reference compensation control structure

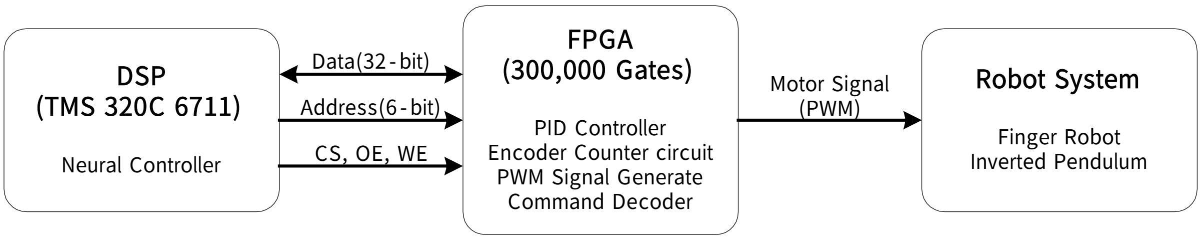

Fig.2. Relationship between each block

Here we try to implement neural network control algorithm on the DSP board. Fig. 2 shows the block diagram of interface structure between each module of the intelligent controller. DSP communicates with FPGA to give compensated signals and the FPGA calculates errors to form PID controllers and then generates PWM signals to motor drivers.

3. PID CONTROLLER ON FPGA

FPGA based PID controller consists of a communication block, an encoder counter block, a PID calculation block, and a PWM generation block. Input signals are 32 bit data bus, 6 bit address bus, control signals such as CS, OE, WE, encoder signals, and 25MHz clock. Output signals are PWM signals.

Fig.3 shows the block diagram of inside of the PID controller.

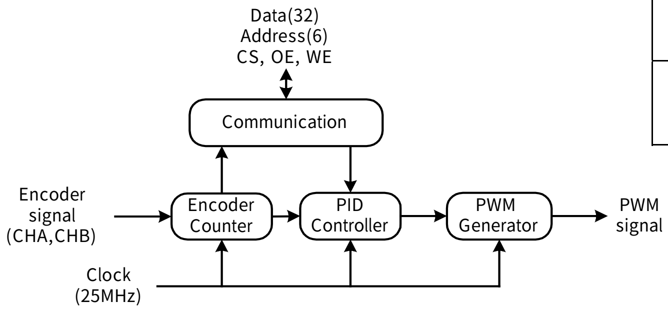

Fig 3. Inside block diagram of PID controller

The detailed description and function of each block can be found in the paper [6].

4. DSP Board

Commercial general purpose a DSP board TMS 320C 6711 is used in this implementation. In order for a DSP to communicate with FPGA PID controllers, 32 data bus is used to share data. Table 1 shows the read/write commands specified with different addresses. In order to distinguish each command, different address pins are used. In this case, we can access 4 PID controllers with respect to each address pin.

At every sampling time, the DSP board has to give compensation values to the FPGA so that the FPGA can add those values to PID controllers. This kind of process can be done by communication between the DSP and the FPGA. Since a whole calculation has to be done in one sample, the DSP and the FPGA have to be synchronized.

FPGA Initialization : Initialize internal variables in FPGA (ex, encoder counter)
PID controller gain : Select PID controller gains in FPGA.
Reference input : Determine reference input value at every sampling time
Neural network output : DSP outputs compensation signals to an FPGA to be added with PID controllers.
Read encoder values: Read encoder values and form errors.

Table 1. Instructions by decoding addresses

	A3	A2	A1	A0	
Write	0	0	0	1	P gain write
	0	0	1	0	I gain write
	0	0	1	1	D gain write
	0	1	0	0	Motor enable
	0	1	0	1	Motor disable
	0	1	1	0	Desired value write
	0	1	1	1	FPGA reset
	1	0	0	0	Neural network output write
Read	0	0	0	1	Read encoder counter read, Initiation of PID control

5. NEURAL NETWORK CONTROLLER

5.1 Reference compensation technique

In this paper, we are implementing on line learning algorithm for neural network. The reference compensation technique has been proposed by Jung and Hsia [2]. The idea of RCT is that neural network compensates at input level by modifying input signals. The same objective function of feedback error learning method can be minimized in on line fashion [3].

The angle error is formed as

$$e_\theta = \theta_d - \theta \qquad (1)$$

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

PID controller for angle control is defined as

$$u_\theta = k_{P\theta} e_\theta(t) + k_{I\theta}\int e_\theta(t)dt + k_{D\theta}\dot{e}_\theta(t) + k_{P\theta}\phi_1 + k_{I\theta}\phi_2 + k_{D\theta}\phi_3 \qquad (2)$$

where ϕ_1, ϕ_2, ϕ_3 are neural network outputs.

In case of inverted pendulum control, cart position is controlled as well as pendulum angle. The position tracking error is formed as

$$e_x = x_d - x \qquad (3)$$

where x_d, x are desired cart position and actual cart position, respectively.

PID control for a cart can be formed as

$$u_x = k_{Px} e_x(t) + k_{Ix} \int e_x(t) dt + k_{Dx} \dot{e}_x(t) + k_{Px}\phi_4 + k_{Ix}\phi_5 + k_{Dx}\phi_6 \quad (4)$$

where ϕ_4, ϕ_5, ϕ_6 are neural network outputs.

The overall control input for the inverted pendulum system

$$u = u_x + u_\theta \quad (5)$$

Fig. 4 shows the control block diagram for robot hand control.

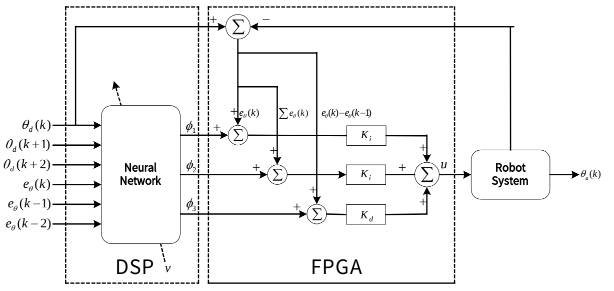

Fig.4. Control block diagram for robot hand

5.2 Neural network learning

For neural network, we have used a general feed-forward structure that has input layer, hidden layer , and output layer. 6 hidden units and 3 output units are used.

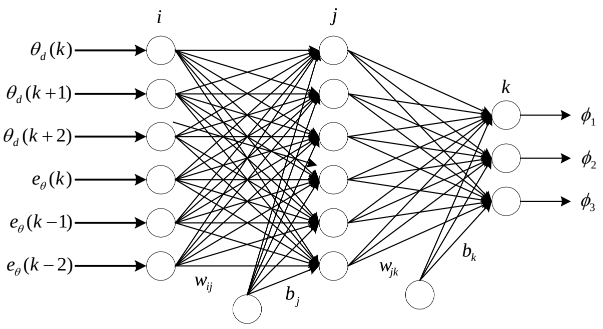

Fig.5. Neural network structure

For nonlinear function at hidden layer and output layer we have used a hyperbolic tangent function as

$$f(x) = \frac{1-\exp(-x)}{1+\exp(-x)} \quad (6)$$

For inverted pendulum control, a general purposed feed-forward neural network is also used, but different number of units are used. The numbers of input layer, hidden layer, and output layer are 12, 12, 6 respectively.

Here, neural network learning algorithm is derived. Since we are doing on-line learning and control, selecting the training signal is very important. Neural network outputs are defined as

$$\Phi = k_P\phi_1 + k_I\phi_2 + k_D\phi_3 \quad (7)$$

If $f(\theta, \dot{\theta}, \ddot{\theta})$ is the system dynamics, equation (4) and (5) can be represented as follows:

$$k_P e + k_I \int e dt + k_D \dot{e} = f(\theta, \dot{\theta}, \ddot{\theta}) - \Phi \quad (8)$$

If the left side of (9) becomes zero , then $\Phi \cong f(\theta, \dot{\theta}, \ddot{\theta})$. This means that inverse dynamics control can be achieved. So here we define the training signal of neural network as

$$v = k_P e + k_I \int e dt + k_D \dot{e} \quad (9)$$

The objective function is defined as

$$E = \frac{1}{2} v^2 \quad (10)$$

Differentiating (11) with respect to weight, we have

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial v}\frac{\partial v}{\partial w} = v\frac{\partial v}{\partial w} = -v\frac{\partial \Phi}{\partial w} \quad (11)$$

Here, we have

$$\frac{\partial \Phi}{\partial w} = K_P \frac{\partial \phi_1}{\partial w} + K_I \frac{\partial \phi_2}{\partial w} + K_D \frac{\partial \phi_3}{\partial w} \quad (12)$$

The update equation in back propagation algorithm is

$$\Delta w(t) = \eta \frac{\partial \psi}{\partial w} v + \Delta w(t-1) \quad (13)$$

$$w(t+1) = w(t) + \Delta w(t) \quad (14)$$

In the similar way, learning can be done for inverted pendulum system.

5.3 Hardware implementation of neural network controller

We used a DSP board of TI TMS320C6711and FPGA of Altera APEXⅡ EP20K300EQC240. Fig. 6 shows real figure of the controller.

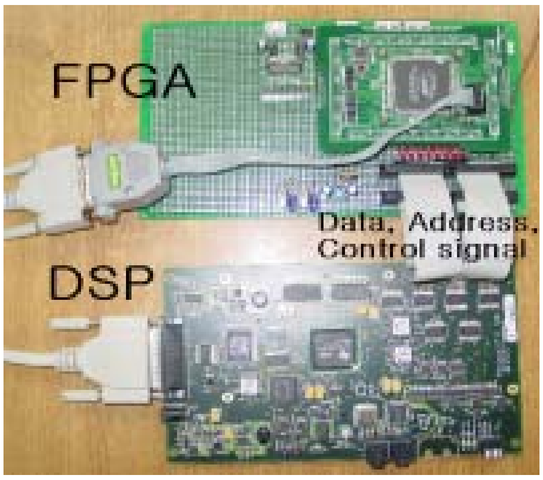

Fig. 6. DSP-FPGA Hardware

6. EXPERIMENTS

Here two experiments are conducted. One is to control robot hand and another experiment is to control inverted pendulum.

6.1 Robot hand

Robot hand is shown in figure 7. It has three fingers and each finger has three joints, but only two joints are actuated. D-H parameter of the robot hand is listed in table 2 .

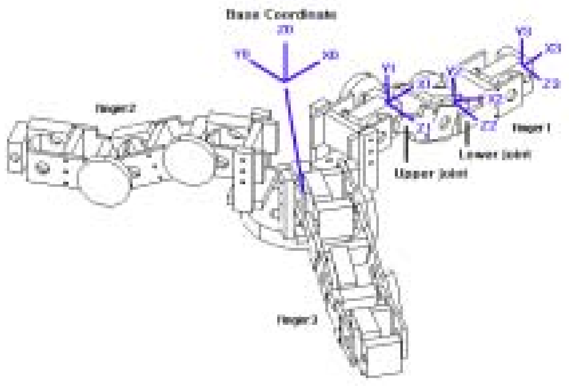

Fig. 7. Robot hand

Table 2. D-H Parameters of robot hand

Joint		θ_i	d_i	a_i	α_i
Finger1	1	0	37.5	90	62
	2	-25	0	0	55
	3	25	0	0	55
Finger2	1	120	37.5	90	62
	2	-25	0	0	55
	3	25	0	0	55
Finger3	1	240	37.5	90	62
	2	-25	0	0	55
	3	25	0	0	55

Figure 8 shows the position tracking by neural network controller with PID controller. Result of the same tracking experiment by PID controller is shown in Figure 9.

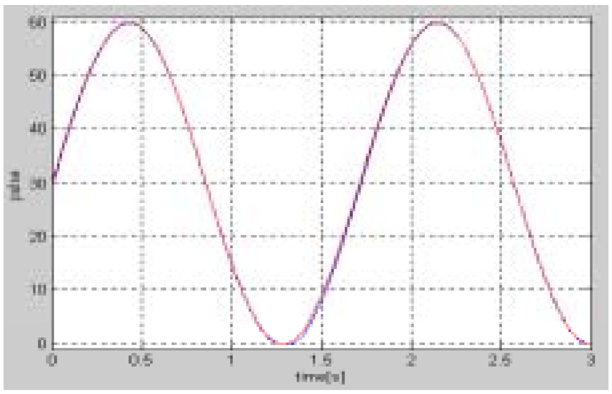

Fig.8. Position tracking by neural network controller

Tracking performances of two controllers are not compared with ease. So the tracking errors are plotted separately in figure 10.

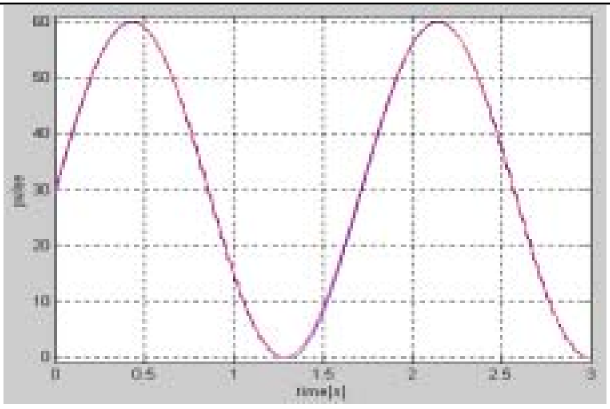

Fig. 9. Position tracking by PID controller

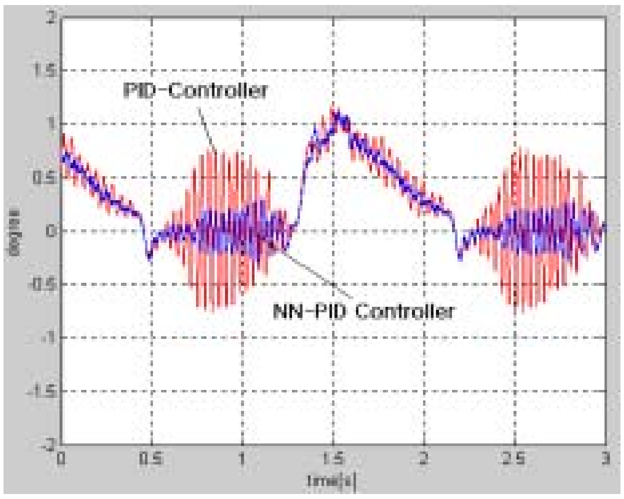

Fig.10. Comparison of performances of PID and neural network controller

From figure 10 we clearly see that position tracking error is reduced when neural network controller is used.

6.2 Inverted pendulum

Next experiment is to control the angle of pendulum and the position of cart simultaneously. Figure 11 shows the experimental setup of the inverted pendulum system The system consists of an inverted pendulum, hardware controller, and a PC.

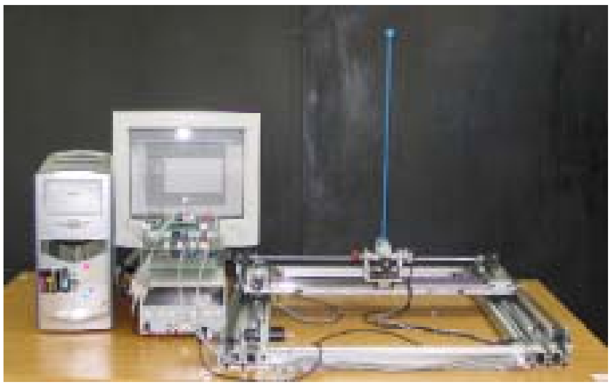

Fig. 11. Inverted pendulum system

Optimized PID controller gains and neural network parameters are listed in Table 3.

Table 3. Controller gains

Gains		Values
Angle control	k_p	-5.5
	k_i	0.002
	k_d	-1.3
Position control	k_p	-1.5
	k_i	-0.05
	k_d	-1.3
Learning rate (η)		0.04
Momentum (α)		0.3

6.2.1 Balancing control

6.2.1.1 PID controller

Figure 12 shows the pendulum error by PID controller. The pendulum is well maintained at balance.

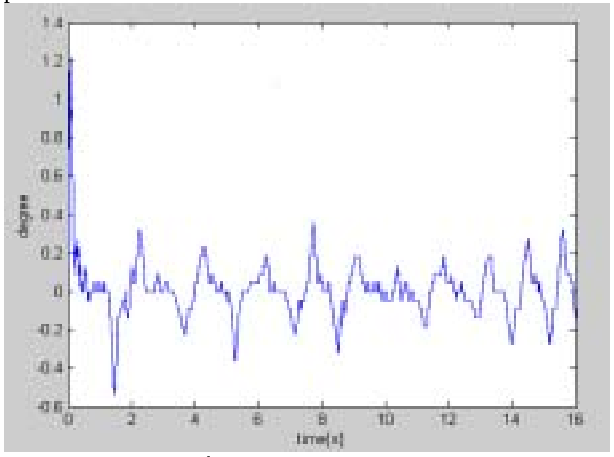

Fig. 12. Angle of the pendulum

However, the cart keeps moving toward one direction as shown in figure 13.

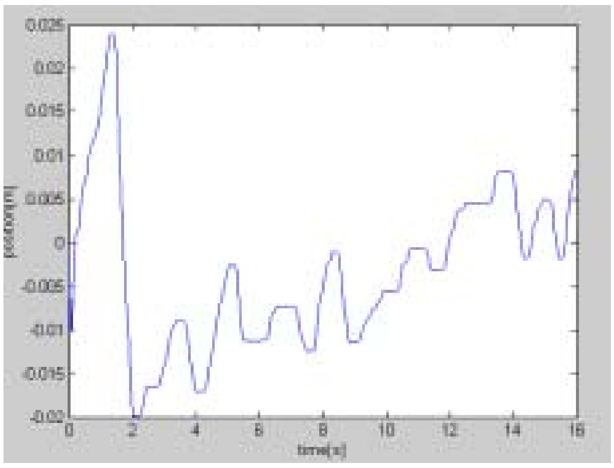

Fig. 13. The position of a cart

6.2.1.2 Neural network controller

The same experiment has been done by neural network controller. The pendulum is required to move toward desired point. Figure 14 shows the angle of the pendulum. Overshoots are observed when external hits by a human are introduced. When external disturbances are present the controller is robust enough to maintain balancing.

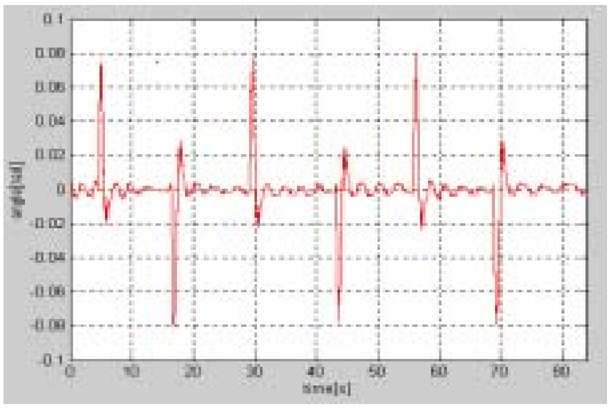

Fig.14. Pendulum Angle by neural network controller

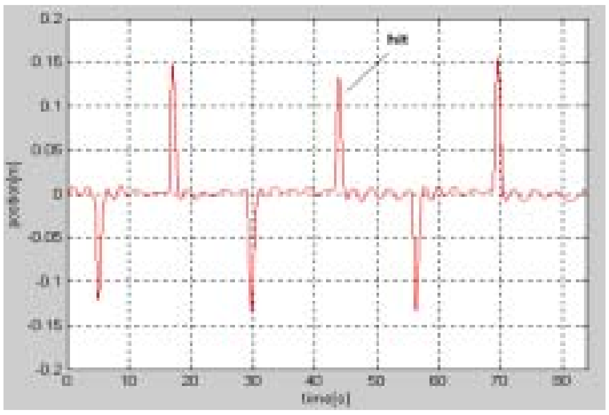

Fig.15. Cart position by neural network

6.5.2 Position tracking

Another interesting experiment has been conducted to test performance of the proposed controller. The cart is required to track a desired sinusoidal trajectory given as $x(t) = 0.15\sin\frac{t}{2}$ while balancing the pendulum. Fig 16 shows the angle of the pendulum.

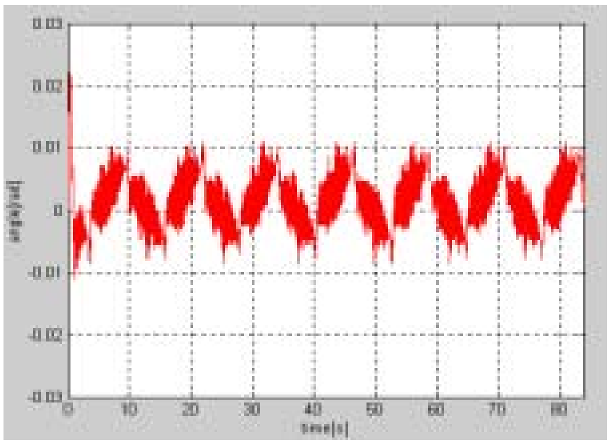

Fig.16. Pendulum angle when T= 4π sec

Fig. 17 shows the position tracking of the cart. We note that small overshoots can be observed when moving direction is changed.

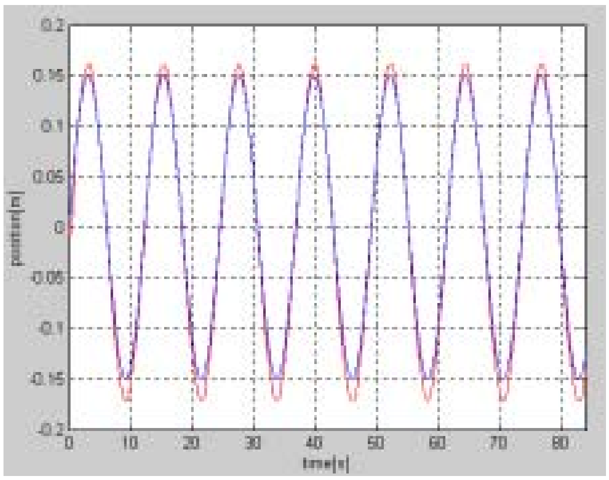

Fig.17. Position tracking of the cart when T = 4π sec

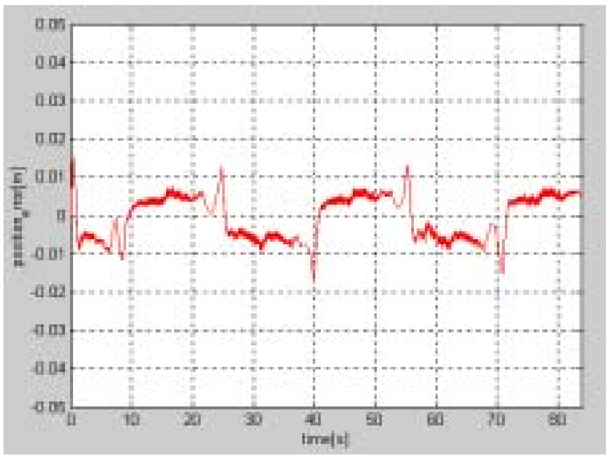

Fig.18. Positional errors when T= 4π sec

In these experiments, pendulum can be balanced within 0.007rad angle error and the cart tracks desired trajectories within the error of 1 ㎝.

6. CONCLUSIONS

This paper addresses the hardware implementation of neural network controller with an FPGA based PID controller. The proposed controller was tested by performing control of robot hand and an inverted pendulum. The proposed controller improves tracking errors of robot hand and performs successfully position tracking of the cart while balancing of the pendulum. Experimental studies confirmed that the proposed controller can be used for any nonlinear systems whose output signals are available since output signals are used to form errors to train neural network on line.

REFERENCES

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

[2] S. Jung and T. C. Hsia, "Neural network Inverse Control Techniques for PD Controlled Robot Manipulator", *ROBOTICA*, pp. 305-314, vol. 19, No 3, 2000

[3] 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

[4] Seul Jung and Sun Bin Yim "Experimental studies of neural network control technique for nonlinear systems", pp. 918-926, vol. 7, no. 11, 2001

[5] H. T. Cho and S. Jung, "Balancing and Position control of an Inverted pendulum on an X-Y Plane using Decentralized neural networks", *Proceedings of the 2003 International Conference on Advanced Intelligent Mechatronics*, pp. 181- 186, 2003.

[6] Sungsu Kim and Seul Jung, "Development of a General Purpose Motion Controller Using a Field Programmable Gate Array", to appear in ICCAS 2003

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

[8] M. Cristea, J. Khor, M. McCormick; FPGA fuzzy logic controller for variable speed generators**,** *Proceedings of the 2001 IEEE International Conference on Control Applications*, pp. 301 304, 2001.

[9] Abdelkrim K. Oudjida et al, A reconfigurable counter controller for digital motion control application, *Microelectronics Journal*, vol. 28, no. 6-7, 1997.

[10] F. Thomas et al, Design and implementation of a wheel speed measurement circuit using field programmable gate arrays in a spacecraft, *Microprocessors and Microsystems*, pp. 553-560, 1999.

[11] Oh et al, "Design of a biped robot using DSP and FPGA", *Proceeding 2002 FIRA Robot World Congress*. 698-701.

[12] A. Kongmunvattana, P. Chongstivatana, A FPGA-based Behavioral Control System for a Mobile Robot, *The 1998 IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 759-762, 24-27, Nov, 1998.