Microprocessor-based Neural Network Controller of a Stepper Motor for a Manipulator Arm

GEORGE K. ADAM[1], GEORGIA GARANI[1],
VILEM SROVNAL[2], JIRI KOZIOREK[2]

[1]Department of Informatics and Telecommunications
Technological Educational Institute of Larissa
41110 Larissa
GREECE
gadam@teilar.gr, http://www.cs.teilar.gr/gadam

[2]Department of Measurements and Control
VSB Technological University of Ostrava
17. listopadu, 15 708 33 Ostrava
CZECH REPUBLIC
vilem.srovnal@vsb.cz

Abstract: - The development and implementation of an FPGA neural network control system for motion control is a task that requires techniques and methods from several engineering fields. In this paper is presented the design of a microprocessor-based neural control system for the realization of the control of a stepper motor for a manipulator arm. The application circuit developed is a specific design of a low-cost embedded system. Simulation and analysis tests were carried out to verify the design of the prototype circuit board. The preliminary results obtained provide the basis for further future research and refinement of the controller.

Key-Words: - Microprocessor control, Neural network, VHDL Model.

1 Introduction

In modern industrial systems, FPGA-based intelligent controllers play an important role in improving the performance of the control system applications. In addition, the rapid development of digital techniques and logic synthesis methods that has been observed the last decade [1], provide more effective implementation of the control circuits design. The expansion of these techniques includes also a number of methods and techniques from artificial intelligence (AI) field. Expert systems and neural networks are some of the artificial intelligence techniques used in computer emulation of human thinking applied in control [2], [3]. Particularly interesting are also fuzzy systems [4] which have had rapid growth in the field of intelligent control (fuzzy control) [5].

Artificial intelligence has made significant advances. Interest in artificial neural networks (ANN) surged in 1985, following the discovery of an effective learning algorithm (back propagation algorithm) [6]. A neural network is a system made up of several basic entities (neurons) which are interconnected and operate in parallel transmitting signals to one another in order to achieve a certain processing task, while back propagation is used to train the network [7]. This algorithm is a gradient method aiming to minimize the total operation error of the neural network by altering the connection weights. In recent years, neural solutions have been suggested for many industrial control systems using mainly feed-forward (or layered) and inverse (or recurrent) configuration neural networks [8], [9]. In inverse configuration, the neural network receives the output of the system under control, and generates an approximation of the input vector, the difference (error) of which is minimized during the training. It can be shown that whatever problems can be solved by inverse network can also be solved by the equivalent feed-forward network with proper external connection. In other words, a neural controller could be build with an input consisted of the system's output plus the output reference, an approach followed in this work too.

This paper presents the design work of a hybrid neural network control system for a manipulator arm, based on Intel 80C188EB microprocessor (25 MHz, 20bits address bus, 1M address space, 8 bits

data bus, 16 I/O pins, 3 timers/counters, 5V). The neural network controller is used in conjunction with a programmable peripheral interface unit to drive a stepper motor that controls the position of the end-tool of the manipulator's arm. The design of the neural controller is based upon a new methodology for implementing neural networks into digital hardware presented by M. Cirstea et. al. [10], and applied in current and speed control of induction motors [11], [12]. This methodology is very interesting and seems to have the potential for a variety of future applications in control systems. In this paper, this neural network implementation design method is applied in the control of a stepper motor. Stepper motors that feature unipolar drives are widely used in applications that require high torque loads and fast position attainment.

The controller basically consists of the 80C188EB microprocessor unit, an 82C55 programmable peripheral interface unit (PPI), an analog-to-digital converter ADC0804, and a neural network controller of manipulator's stepper motor, implemented in a Xilinx XC2018 FPGA (Field Programmable Gate Array) device. The control system developed is an open modular architecture, which allows the incorporation of additional modules particularly communication modules for distributed control. Verification of the design is carried out through simulations in VHDL (IEEE Standard Hardware Description Language), and real tests in a machines construction company (*Adam Machines Constructions Co., Volos, Greece*).

The remainder of this paper is organized as follows. Section 2 provides a brief description of the manufacturing workcell and the manipulator. Section 3 describes the methodology used in designing the neural controller for the stepper motor of the manipulator under control. Section 4 provides details of the hardware design implementation process, including VHDL model generation of the motor neural controller and FPGA synthesis. In section 5 is provided a brief performance analysis of the simulation results obtained for the controller's functionality. Conclusions and related future research are given in Section 6.

2 The Manufacturing Workcell

The initial manufacturing system under investigation was a robotics workcell that was consisted of an in-house developed manipulator-arm, the controller unit (based on Motorola MC68705P3 microcontroller), an operator's control panel (keyboard & indicators), and various other mechanical parts and equipment required for the assembly tasks and experiments performed (e.g., a working platform and a conveyor belt). The microcontroller MC68705P3 is a single-chip integrated circuit designed for embedded industrial control applications. A simplified block diagram of the microcontroller-based manipulator controller is given in Fig.1.

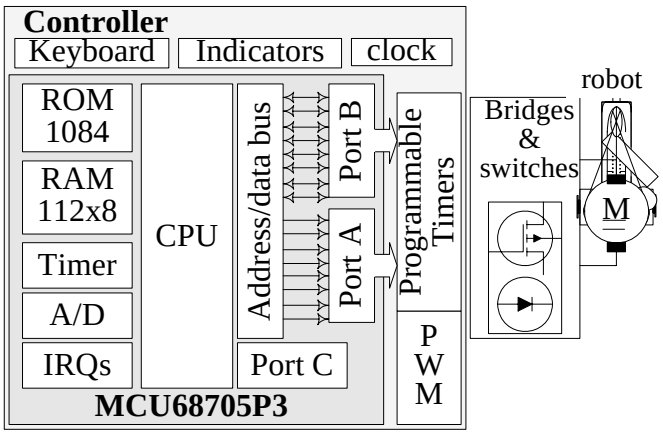

Fig. 1. Block diagram of the controller.

The new proposed architecture of the neural network controller is based upon the 80C188EB Intel microprocessor, designed for embedded industrial control applications and the FPGA implemented neural network. This work is focused on the control system that drives a single stepper motor, which in turn defines the angular position of the manipulator's end-tool. A configuration of the control system is shown in Fig. 2.

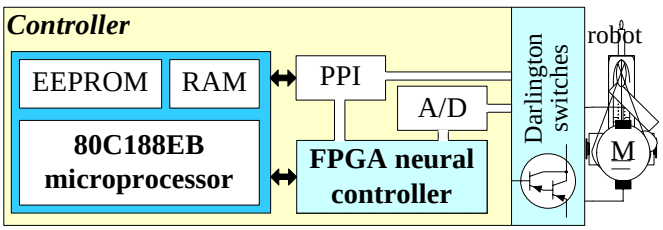

Fig. 2. Block diagram of the hybrid control system.

2.1 The Manipulator

The developed in-house manipulator is a joint-arm mechanical system with three parallel vertical axes driven by DC servo motors (each for a single axis), and a roll axis (at the end-tool) driven by a stepper motor (four degrees of freedom), that allow rotational and linear movements along the X, Y and Z axis. The manipulator arm is driven by small electrical DC motors. The manipulator basically is designed to perform frequent assembly tasks in a vertical manner, in other words, execute movements up and down along the Z-axis, which are servo-controlled. However, straight linear movements can

also be programmed and executed, however within a specified area. In this case, the manipulator can be programmed to perform coordinate transformation of the world coordinates (X, Y, Z) of the end-tool into a number of corresponding joint coordinates (θ_1, θ_2 arms, and a end-tool's rotation angles) of manipulator's arm.

Accuracy is required in the manipulation of materials, for this reason a four-coil stepper motor (of permanent-magnet type, 12V supply) is used to drive the end-tool as the acting tool of the manipulator. A stepper motor is preferred, because the position of the end-tool (a simple mechanic gripper) can be controlled precisely (0.9^o step). It is also important to consider that the most optimum arrangement of the joints for best manipulation of objects and tasks performance could be calculated by using the Jacobian matrix, which for every joints arrangement represents the relation between the joints displacements and the current position and orientation of the end-tool. In other words, defines the linear transformation form joint co-ordinates to Cartesian co-ordinates.

Let $r = [x\ y\ z\ a]^T$ be the position vector, where $[x\ y\ z]^T$ is the position of the of the acting tool and a is the rotation angle to the z axis. In that case the Jacobian matrix is:

$$J = \begin{bmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} & 0 & 0 \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \quad (1)$$

where l_1, l_2 are given manipulators arm lengths, s_1 is $\sin(\theta_1)$, s_{12} is $\sin(\theta_1-\theta_2)$, c_1 is $\cos(\theta_1)$, and c_{12} is $\cos(\theta_1-\theta_2)$.
An estimation of the measurement (b_m) for optimum manipulation is given by the following equation:

$$bm = \left|\det J\right| = l_1 l_2 \left|s_2\right| \quad (2)$$

As a result, for given lengths l_1, l_2, and joint variables θ_1, and θ_2, an optimal configuration is achieved for $\theta_2 = \pm 90^o$.

The positioning of the end-tool is particularly important for execution of specific assembly tasks. For this reason, the task of the neural network controller is to analyze the input data from the stepper motor driver and generate information (triggering signals) required for the optimum operation of the manipulator arm.

3 Neural Network Controller

The architecture of the neural network controller is basically a parallel input-output system where computation is performed in a distributed manner. Such a system is usually ideally implemented using a FPGA device large enough to implement relatively complex functions (such as motor rotor angular position) on a single chip.

3.1 The Stepper Motor

The design of the neural controller is focused in the control of a stepper motor (Astrosyn, SST0009) that drives the angular position of the end-tool. The stepper motor is with a polyphase stator (four-phase stator winding). The motor does not require a three-phase supply, but 12V. This motor is designed to rotate a specific number of degrees for every electric pulse received by the control unit. In other words, is like a digital motor because it is moved in discrete steps as it traverses from 0.9^o per step (high-precision) through 360^o.

The stepper motor is driven by using NPN Darlington amplifier pairs to provide a large current to each coil. The operation of the stepper motor is easily controlled by the voltage applied in the stator that forces the rotor to rotate [13]. For example, if a dc voltage is applied to one phase of the stator, this will cause current to flow in the phase producing a stator magnetic field that will interact with the rotor magnetic field inducing a counterclockwise torque on the rotor, and causing the rotor to line up with the new position of the magnetic field. By continuing this pattern with the other phases it is build a table (stored in a look-up table) showing the rotor as a function of the voltage applied to the stator of the motor. In this way a control signal (output voltage) is produced as a series of control pulses, each one of which controls the rotor position by advancing the stepper motor with a certain degree. The number of mechanical degrees θ_m moved per step depends on the number of poles P, and corresponds to given number of electrical degrees θ_e given by the following equation:

$$\theta_m = \frac{2}{P}\theta_e \quad (3)$$

Using Eq. (3) the speed of the stepper motor can be related to the number of pulses per unit time into its neural controller. If we differentiate both sides of this equation with respect to time, then we obtain the following relationship between the electrical ω_e and mechanical ω_m rotational speed of the motor:

$$\omega_m = \frac{2}{P}\omega_e \tag{4}$$

In other words, the speed of the motor in revolutions per unit time is related to the number of pulses.

This technique simply applies the voltage to each of the four coils in the proper order to achieve the desired rotation. When a voltage is applied, the stepper motor moves a few degrees of rotation with each pulse of current. One step pulse is required for every step of the motor shaft.

3.2 The Neural Controller

Based on this voltage-current control strategy a neural controller of the stepper motor has been created and implemented in a FPGA unit using a VHDL model. The neural stepper motor controller developed uses a FPGA alongside with a programmable peripheral interface unit 82C55 (PPI) to control the stator voltage and current induced. A simplified block diagram of the FPGA neural controller configuration is shown in Fig. 3.

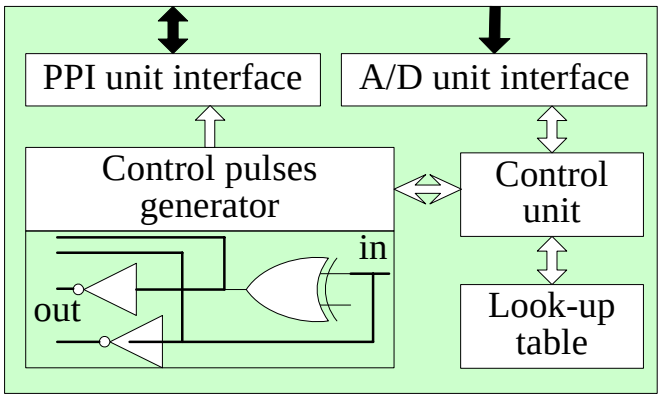

Fig. 3. Block scheme of the FPGA neural controller.

The mathematical model of the neural controller for the stepper motor consists of a set of m x n matrices containing the parameters of the neurons in the neural network. Each matrix refers to one neural layer and each row in a matrix contains the parameters of a single neuron. Each neuron has several inputs, where each input is characterized by a certain weight indicating the influence of the corresponding signal over the neuron output. The first elements of a row are the neuron weights while the last one is the threshold level. For example, the motor's matrix is represented by an 8 x 10 matrix of inputs consisting of logical 0s and 1s that correspond to the signals that turn on and off the voltage pulses (current signals) that drive the motor. Each line in the matrix contains a set of quantities (currents and voltages) that characterize the stepper motor operation at a certain moment in time. This input vector of 80 signals is connected to the respective 90 neurons at the input layer. In other words, each neuron accumulates all the input weighted signals which then passes to the output through a transfer function of threshold type (output is +1 if the input exceeds a threshold value). The number of layers and the number of the neurons in each layer depends on the complexity of the problem. In our case, the sum of the network weights is about 850 altogether.

The operation of this neural network controller can be described in terms of logic functions and operations using basic AND, OR and NOT interconnected logic gates. The neural controller actually outputs signals that control the angle of rotor rotation (given as the number of electrical degrees θe related to the number of pulses per unit time, around 400: $360^o/0.9^o$, and the angular speed of the rotor ωm), required to establish the precise position of the end-tool.

As a means of communication and control of the 80C188EB-based overall system controller, an application control program in Intel 80x86 family assembly language is provided. The software module that was implemented in the controller generates, through the 82C55 programmable interface adapter, the triggering signals to the drive (a high-voltage and high current ULN2004 circuit) of the stepper motor that positions the manipulator's end-tool.

4 Hardware Implementation

The hardware development is focused in selecting the appropriate commercially available components to build the control system according to the configuration and specifications of the controller. Hardware implemented neural networks are essentially arrays of interconnected digital processing structures that operate concurrently. Digital technology provides basic building blocks for constructing neural networks as ASIC or FPGA implementations. Actually the overall controller implementation is a single-board microcomputer system.

4.1 VHDL Model Generation

The generation of the VHDL model of the digital artificial neural network, that describes the hardware implementation, is accomplished in three stages by using the universal C++ programs provided by M. Cirstea et. al. at their book [10]. First the mathematical description (matrix description) of the

artificial neural network is converted to a netlist description that contains the circuit nodes and gates. Then this description is optimized by eliminating the redundant components and finally converted to VHDL description of the hardware implemented neural network. This file contains an architecture that comprises a number of internal signals and a list of assignment statements that model several identical logic gates by associating a logical expression with an internal or an output signal. Further details about these software programs could be found on the above reference.

A fragment of the VHDL code generated for the stepper motor neural controller is given below.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY network1 IS
  PORT(d_in : IN std_logic_vector(-1 DOWNTO 0);
       d_out: OUT std_logic_vector(-1 DOWNTO 0));
END network1;
ARCHITECTURE arch_network1 OF network1 IS
port_id: integer:=1; delay:real:=0.10;
signal step: integer:=1; msb_steps:integer;
variable adc_value:integer:=10; ….
BEGIN
constant p:real:=3.14; ….
END arch_network1;
CONFIGURATION conf_network1 OF network1 IS
  FOR arch_network1
  END FOR;
END conf_network1;
```

In the above VHDL neural model the stepper motor is an entity having an input port (the stator voltage) and an output port (the rotor angular position). Simulations were performed using ModelSim software package (ModelSim SE Plus 5.7d, Model Technology – Mentor Graphics Corporation) to validate and refine the controller design.

4.2 Synthesis

Once the simulated VHDL model was corrected and refined the obtained VHDL file was further synthesized using Leonardo Spectrum software package (*Mentor Graphics, Inc.*), a tool for synthesis, simulation and testing of FPGA implemented circuits. A schematic design of the controller is shown in Fig. 4.

The specific neural controller was implemented into a Xilinx XC2018 FPGA unit. FPGA devices are widely used in embedded systems. The specific device was chosen for easy development and upgrade of the previous control system (that was based on MC68705P3). The compilation of this controller with the VHDL components required about 850 logical elements (logic gates) (45% of the total amount of logical elements of the XC2018 FPGA).

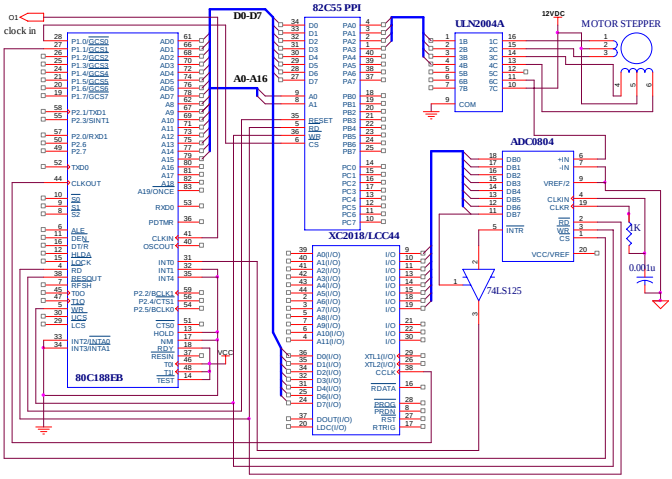

Fig. 4. Schematic design of the microprocessor-based FPGA neural controller.

The ULN2004 Darlington array is used to drive the stepper motor. The 12 volts are fed to the stepper motor through an integrated circuit of Darlington arrays that contains Darlington configurations of power transistors, which actually switch the 12 volts supply on and off. The system's synchronization is accomplished through an external (quartz) pulse generator (32MHz), in order to ensure high accuracy and TTL level signal required. For simplification is not shown the memory units (RAM and EPROM) and the latch circuits (74HC573) that buffer the data and addresses from the microprocessor towards the memory units and the peripheral devices.

The controller utilizes the 82C55 PPI to perform the control and realize the generation of control signals (pulses), through look up tables of values stored in the computer's EPROM memory, which are compared to those in the neural controller (corresponding to the required voltages). The controller uses the address bits A0-A1 to access the programmable unit and provide the drive signals to control the interfaced motor. The 82C55 PPI init can interface any TTL-compatible I/O device to the microprocessor and provide a maximum of 4mA of current at each output.

Since the FPGA unit does not have an analog to digital converter to receive the motor current measurements (voltage pulses) an external ADC0804 unit was employed. The analog-to-digital converter (ADC0804) converts the input voltage from the stepper motor driver into a digital code sequence. It enables the measurement of the output voltage in the range from 0 to Vcc. The interrupt signal (INTR) is connected to an interrupt input of the 80C188EB microprocessor to signal periodically

the end of this conversion. This code sequence of bits (e.g., 11,10,01,00) is used by the neural controller to estimate the pulses codes generated and provided to the PPI interface unit. The reference voltage is generated internally in the microprocessor and is used to compare the digitized value of the voltage on the ADC output. Depending on the ADC calculation an appropriate value is chosen from the neural look-up table and loaded into the PPI unit. This scheme in conjunction with the ULN2004 circuit is used to provide the prerequisite amount of current (voltages) to the windings of the stepper motor in a predefined sequence of pulses. In this way, this pulsed switching of the voltage in the four windings is achieved by providing on and off periods of voltage supply.

5 Performance Analysis

Simulation results (using ModelSim) of the FPGA implementation of the neural controller have shown a satisfactory sequence of controller's executions. In Fig. 5 is shown a fragment of the simulation waveforms obtained during motor control at the four gate pins of the power transistors in the ULN2004 circuit that drive the stepper motor of the manipulator's end-tool.

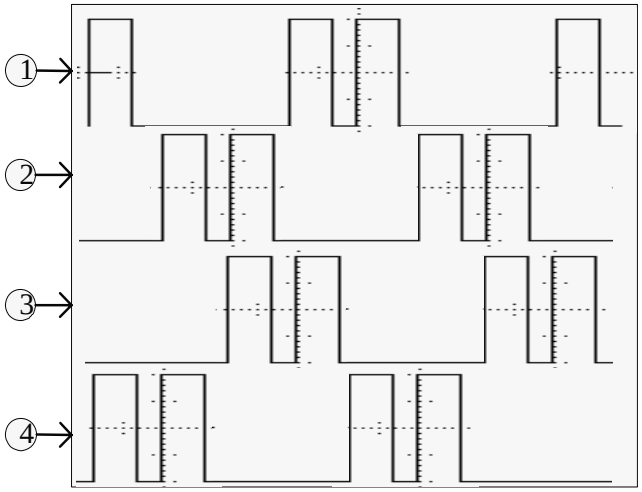

Fig. 5. Full step waveforms in the power transistors.

In addition a set of experiments was carried out to verify the control performance of the drive system based on the entire VHDL model of the controller. The system has been tested in various operations and variations of the load. For instance, in most of the cases the FPGA controller seemed to improve the response of the drive system (to approximately 0.45s from 0.55s) under constant load and revolutions per minute.

6 Future Work and Conclusions

This is a preliminary research which is still under progress, and among other objectives (e.g. improve high-level control), current work is focused in refining and extending the implementation of this hybrid controller to enable distributed processing and control. For this reason the intention is to use it in communication with an external module Rabbit RCM2200, that provides complete full duplex Ethernet 10Base-T port in order to implement XML-RPC mechanisms to perform distributed control of the overall manufacturing workcell.

The development of the current proprietary control circuit has eliminated the expenses of high performance controllers. The design and development of the microprocessor-based control system including an FPGA implementation of the neural controller required significant less time compared to an application specific integrated system. In practice, this hybrid control system was found to be quite applicable in assisting the operation and control of the manipulator's arm. Simulation results have shown that simulation tends to be more time-consuming and debugging more difficult.

References:

[1] T. Sasao, *Logic Synthesis and Optimization*. Norwell, MA: Kluwer Academic Publishers, 1997.

[2] B.K. Bose, "Expert System, Fuzzy Logic, and Neural Network Applications in Power Electronics and Motion Control," *Proc. IEEE Special Issue on Power Electronics and Motion Control*, vol. 82, pp. 1303-1323, 1994.

[3] A.A. Hopgood, *Intelligent Systems for Engineers*, 3rd ed. New York: CRC Press, 2001.

[4] L.A. Zadeh, "Fuzzy Sets," *Information and Control*, vol. 8, pp. 338-353, 1965.

[5] M.J. Er, and N.E. Mastorakis, "Fuzzy Control of Robotic Manipulators," *Journal of Applied Mathematics and Computer Science*, vol. 7, no. 3, pp. 611-637, 1997.

[6] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, pp. 533-536, 1986.

[7] D.A. White, and D.A. Sofge, *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*. New York: Multiscience Press, 1992.

[8] W.T. Miller, R.S. Sutton, and P.J. Werbos, *Neural Networks for Control*. Cambridge MA: MIT Press, 1992.

[9] G.W. Irwin, K. Warwick, and K.J. Hunt, *Neural Network Applications in Control*. London: IEE Press, 1995.

[10] M.N. Cirstea, A. Dinu, J.G. Khor, and M. McCormick, *Neural and Fuzzy Logic Control of Drives and Power Systems*. Oxford: Newnes, 2002.

[11] A. Dinu, "FPGA Neural Controller for Three Phase Sensorless Induction Motor Drive Systems," *PhD Thesis*, De Montfort University, 2000.

[12] A. Dinu, M.N. Cirstea, M. McCormick, A. Ometto, and N. Rotondale, "Sensorless Motor Control Strategy Optimised for FPGA Hardware Implementation," *Journal of Electrical Engineering*, vol. 1, no. 1, pp. 26-31, 2001

[13] S.J. Chapman, *Electric Machinery Fundamentals*. New York: McGraw-Hill, 1999.