We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Chapter 8

FPGA-Based Motion Control IC for Linear Motor Drive X-Y Table Using Adaptive Fuzzy Control

Ying-Shieh Kung, Chung-Chun Huang and Liang-Chiao Huang

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48457

1. Introduction

The development of a compact and high performance motion controller for the X-Y table of a CNC machine has been an important field in literatures (Groove, 1996; Goto et al., 1996; Hanafi et al., 2003). The typical architecture of the conventional motion control system for X-Y table is shown in Fig. 1, which consists of a central controller, two sets of servo drivers and an X-Y table. The central controller, which usually adopts a float-pointed processor, performs the function of motion trajectory and data communication with servo drivers and with external device. Each servo driver usually use a fixed-pointed processor, some specific ICs and an inverter to perform the functions of position/speed/current control at each single axis of X-Y table and to do the data communication with the central controller. Data communication between two devices uses an analog signal, a bus signal or a serial asynchronous signal. However, the motion control system in Fig.1 has some drawbacks, such as large volume, easy effect by the noise, expensive cost, inflexible, etc. In addition, data communication and handshake protocol between the central controller and servo drivers slow down the system executing speed.

In recent years, the FPGA has been widely applied in implementing the digital control system (Cho, 2009; Monmasson et al. 2011; Sanchez-Solano et al. 2007). Besides, an embedded processor IP and an application IP can be developed and downloaded into FPGA to construct a SoPC environment (Altera, 2004; Hall and Hamblem, 2004), allowing the users to design a SoPC module by mixing hardware and software in one FPGA chip (Kung et al. 2004; Kung and Tsai, 2007; Kung and Chen, 2008). Therefore, based on the FPGA technology, we improve the aforementioned drawbacks and integrate the central controller and the controller part of two servo drivers in Fig. 1 into a motion control IC in this study, which is shown in Fig. 2. Our proposed motion control IC has two IPs (Intellectual Properties). One IP performs the functions of the motion trajectory by software. The other IP

performs the functions of two axes' position/speed/current controllers by hardware. As the results, this two IP will parallel processing in FPGA, and the hardware/software co-design technology in FPGA can make the motion controller of X-Y more compact, flexible, better performance and less cost. Further, the X-Y table usually leads to the existence of unmodelled dynamics and disturbances which often significantly deteriorate the system performance during a machining process. Many studies attempt to improve the tracking performance in a machining process (Lin et al., 2006; Wang and Lee, 1999). Lin et al. (2006) adopts a recurrent-neural-network sliding-mode controller to improve the motion tracking performance of the X-Y table. Wang and Lee (1999) integrate the cross-coupled control and neural network techniques to achieve a high accuracy of the motion tracking in the linear motor X-Y table. However, due to the complicate computation of the neural-network, the algorithms of above two studies are realized in the PC-based control system.

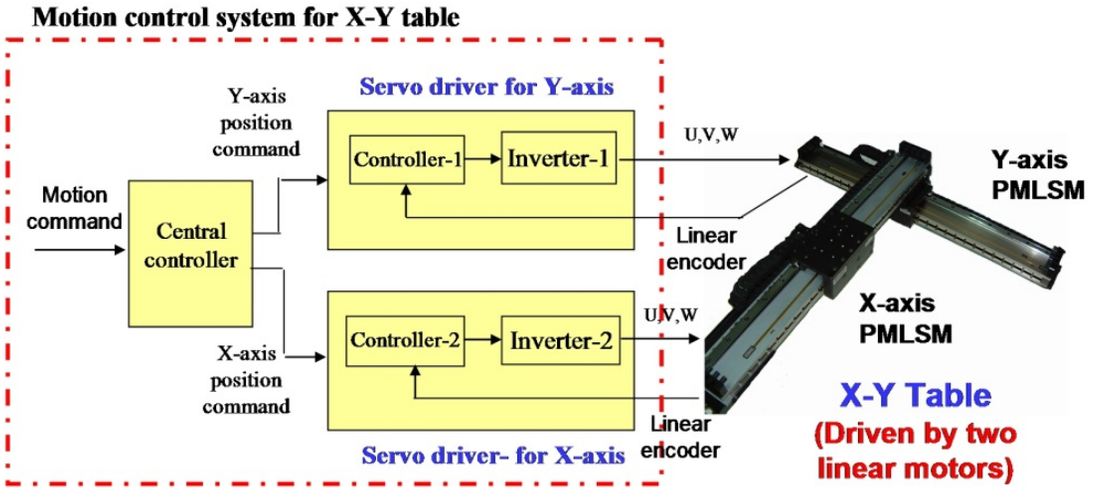

Figure 1. Conventional motion control system for X-Y table

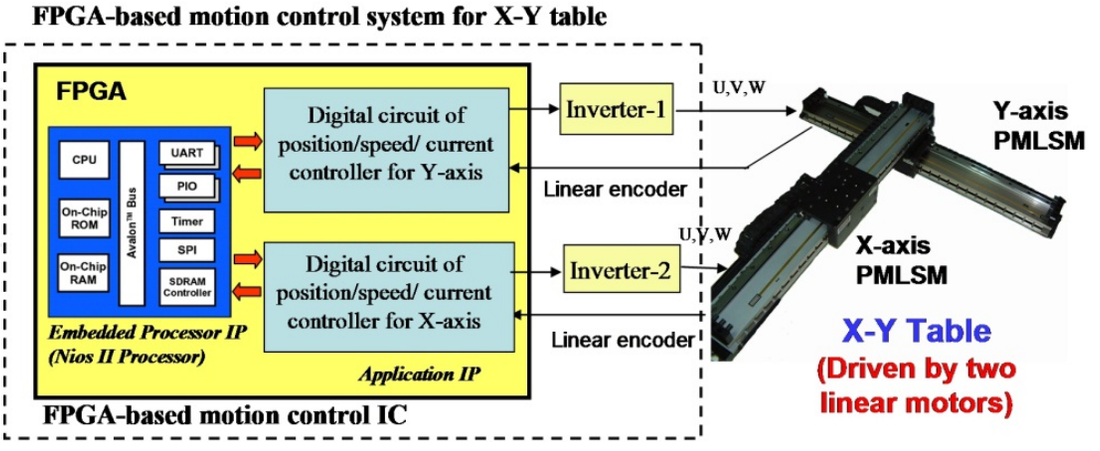

Figure 2. Proposed FPGA-based motion control system for X-Y table

In this chapter, a motion control IC for linear motor drive X-Y table based on FPGA (Field programmable gate array) technology is presented and shown in Fig.3. Firstly, the mathematical model of the X-Y table is defined. Secondly, an adaptive fuzzy controller

(AFC) is introduced and adopted in position loop of X-Y table to improve the motion tracking performance under unmodelled uncertainty condition. Thirdly, in implementation, an FPGA embedded by a Nios II processor is used to design the overall circuits of the motion control IC which the scheme of position/speed/current control for two PMLSMs (permanent magnetic linear synchronous motors) is realized by hardware in FPGA and the motion trajectory algorithm for X-Y table is implemented by software using Nios II embedded processor. To reduce the FPGA resource usage, an FSM (Finite state machine) joined by a multiplier, an adder, a LUT (Look-up table), some comparators and registers is used to model the overall AFC algorithm. And VHDL (VHSIC hardware description language) is adopted to describe the FSM. Herein, Altera Stratix II EP2S60, which has 48,352 ALUTs (Adaptive Look-Up Tables), total 2,544,192 RAM bits, and a Nios II embedded processor which has a 32-bit configurable CPU core, 16 M byte Flash memory, 1 M byte SRAM and 16 M byte SDRAM, is used. Therefore, a fully digital motion controller can be implemented by an FPGA using hardware/software co-design technology which will make the motion controller of the X-Y table more compact, flexible and better performance. Finally, an experimental system is set up to verify the performance of the proposed motion control IC for linear motor drive X-Y table.

2. System description of X-Y table and motion controller design

The internal architecture of the proposed FPGA-based controller system for a linear motor drive X-Y table is shown in Fig. 3, in which the motion trajectory is implemented by software using Nios II embedded processor; the position, speed and current vector controller for two PMLSMs are implemented by hardware in FPGA chip. The mathematical modeling of PMLSM, AFC algorithm and motion trajectory planning are introduced as follows:

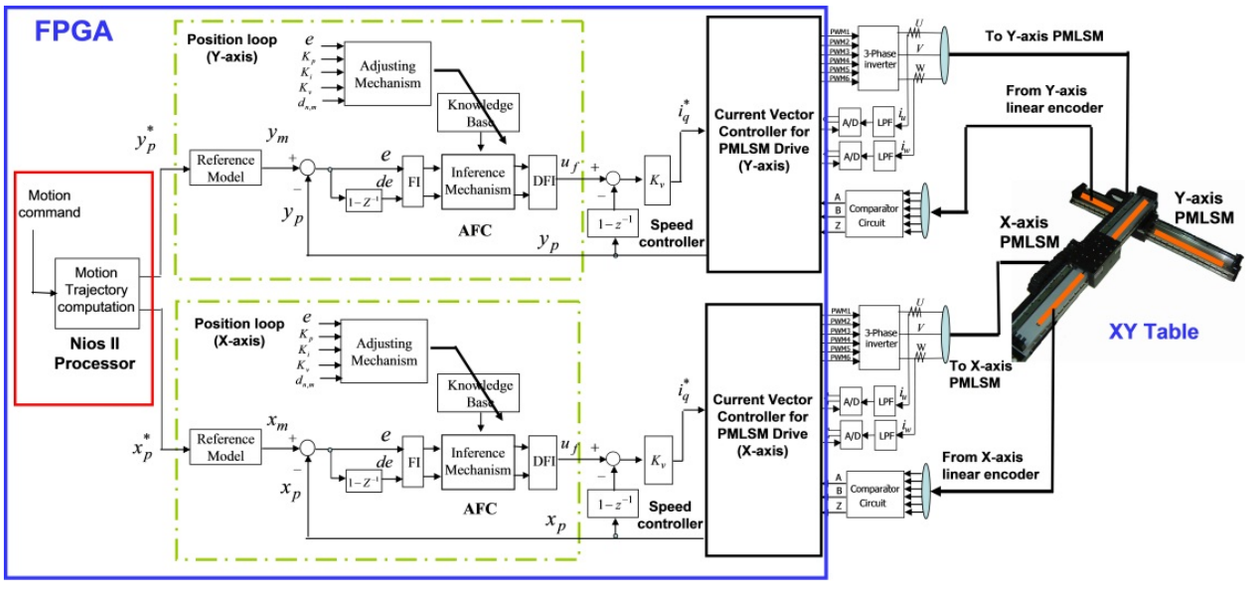

Figure 3. The architecture of a motion controller system for linear motor drive X-Y table

2.1. Mathematical model of the PMLSM drive

The dynamic model of a typical PMLSM can be described in the synchronous rotating reference frame, as follows

$$\frac{di_d}{dt} = -\frac{R_s}{L_d} i_d + \frac{\pi}{\tau}\frac{L_q}{L_d}\dot{x}_p i_q + \frac{1}{L_d}\mathrm{v}_d \tag{1}$$

$$\frac{di_q}{dt} = -\frac{\pi}{\tau}\frac{L_d}{L_q}\dot{x}_p i_d - \frac{R_s}{L_q} i_q - \frac{\pi}{\tau}\frac{\lambda_f}{L_q}\dot{x}_p + \frac{1}{L_q}\mathrm{v}_q \tag{2}$$

where v_d, v_q are the d and q axis voltages; i_d, i_q, are the d and q axis currents, R_s is the phase winding resistance; L_d, L_q are the d and q axis inductance; $\dot{x}_p$ is the translator speed; λ_f is the permanent magnet flux linkage; τ is the pole pitch. The developed electromagnetic thrust force is given by

$$F_e = \frac{3\pi}{2\tau}((L_d - L_q)i_d + \lambda_f)i_q \tag{3}$$

The current control of a PMLSM drive is based on a vector control approach. That is, if we control i_d to 0 in Fig.3, the PMLSM will be decoupled, so that control a PMLSM will become easy as to control a DC linear motor. After simplification and considering the mechanical load, the model of a PMLSM can be written as the following equations,

$$F_e = \frac{3}{2}\frac{\pi}{\tau}\lambda_f i_q \triangleq K_t i_q \tag{4}$$

with

$$K_t = \frac{3}{2}\frac{\pi}{\tau}\lambda_f \tag{5}$$

and the mechanical dynamic equation of PMLSM in x-axis table is

$$F_e - F_L = M_m \frac{d^2 x_p}{dt^2} + B_m \frac{dx_p}{dt} \tag{6}$$

where F_e, K_t, M_m, B_m and F_L represent the motor thrust force, the force constant, the total mass of the moving element, the viscous friction coefficient and the external force, respectively. In addition, the current loop of the PMLSM drive in Fig.3 includes PI controller, coordinate transformations of Clark, Modified inverse Clark, Park, inverse Park, SVPWM (Space Vector Pulse Width Muldulation), pulse signal detection of the encoder etc. The coordination transformation of the PMLSM in Fig. 3 can be described in synchronous rotating reference frame. Figure 4 is the coordination system in rotating motor which includes stationary *a-b-c* frame, stationary α-β frame and synchronously

rotating *d-q* frame. Further, the formulations among three coordination systems are presented as follows.

1. *Clarke* : stationary *a-b-c* frame to stationary α-β frame.

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & \frac{-1}{3} & \frac{-1}{3} \\ 0 & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{7}$$

2. Modified $Clarke^{-1}$: stationary α-β frame to stationary *a-b-c* frame.

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{-1}{2} & \frac{\sqrt{3}}{2} \\ \frac{-1}{2} & \frac{-\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} v_\beta \\ v_\alpha \end{bmatrix} \tag{8}$$

3. *Park* : stationary α-β frame to rotating *d-q* frame.

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{9}$$

4. $Park^{-1}$: rotating *d-q* frame to stationary α-β frame.

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} v_d \\ v_q \end{bmatrix} \tag{10}$$

where θ_e is the electrical angle.

In Fig. 3, two digital *PI* controllers are presented in the current loop of PMLSM. For the example in *d* frame, the formulation is shown as follows.

$$e_d(k) = i_d^*(k) - i_d(k) \tag{11}$$

$$v_{p\_d}(k) = k_{p\_d}\, e_d(k) \tag{12}$$

$$v_{i\_d}(k) = v_{i\_d}(k-1) + k_{i\_d}\, e_d(k-1) \tag{13}$$

$$v_d(k) = v_{p\_d}(k) + v_{i\_d}(k) \tag{14}$$

the e_d is the error between current command and measured current. The k_{p_d}, k_{i_d} are *P* controller gain and *I* controller gain, respectively. The $v_{p\_d}(k)$, $v_{i\_d}(k)$, $v_d(k)$ are the output of *P* controller only, *I* controller only and the *PI* controller, respectively. Similarity, the formulation of *PI* controller in *q* frame is the same.

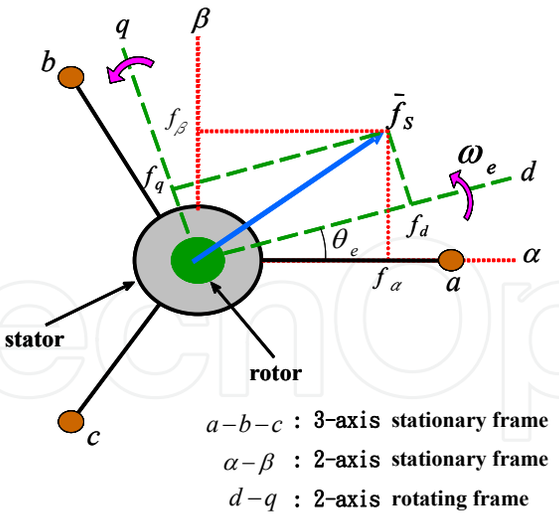

Figure 4. Transformation between stationary axes and rotating axes

2.2. Adaptive fuzzy controller (AFC) in position control loop

The green dash rectangular area in Fig. 3 presents the architecture of an AFC. It consists of a fuzzy controller, a reference model and a parameter adjusting mechanism. Detailed description of these is as follows.

1. Fuzzy controller (FC):

In Fig.3, the tracking error and the change of the error, *e*, *de* are defined as

$$e(k) = \psi_m(k) - \psi_p(k) \tag{15}$$

$$de(k) = e(k) - e(k-1) \tag{16}$$

and *e*, *de* and u_f are input and output variables of FC, respectively. Besides ψ_m represents x_m or y_m, and ψ_p represents x_p or y_p. The design procedure of the FC is as follows:

a. Take the *e* and *de* as the input variables of the FC, and define their linguist variables as *E* and *dE*. The linguist value of *E* and *dE* are {A_0, A_1, A_2, A_3, A_4, A_5, A_6} and {B_0, B_1, B_2, B_3, B_4, B_5, B_6}, respectively. Each linguist value of *E* and *dE* is based on the symmetrical triangular membership function which is shown in Fig.5. The symmetrical triangular membership function are determined uniquely by three real numbers $\xi_1 \le \xi_2 \le \xi_3$, if one fixes $f(\xi_1) = f(\xi_3) = 0$ and $f(\xi_2) = 1$. With respect to the universe of discourse of [-6.6], the numbers for these linguistic values are selected as follows:

$$\begin{aligned} &A_0 = B_0 : \{-6,-6,-4\}, A_1 = B_1 : \{-6,-4,-2\}, A_2 = B_2 : \{-4,-2,0\}, \\ &A_3 = B_3 : \{-2,0,2\}, A_4 = B_4 : \{0,2,4\}, A_5 = B_5 : \{2,4,6\}, A_6 = B_6 : \{4,6,6\} \end{aligned} \tag{17}$$

b. Compute the membership degree of the *e* and *de*. Figure 5 shows that the only two linguistic values are excited (resulting in a non-zero membership) in any input value, and the membership degree is obtained by

$$\mu_{A_i}(e)=\frac{e_{i+1}-e}{2} \text{and } \mu_{A_{i+1}}(e)=1-\mu_{A_i}(e) \tag{18}$$

where $e_{i+1}\triangleq -6+2*(i+1)$. Similar results can be obtained in computing the membership degree $\mu_{B_j}(de)$.

c. Select the initial fuzzy control rules, such as,

$$\text{IF } e \text{ is } A_i \text{ and } \Delta e \text{ is } B_j \text{ THEN } u_f \text{ is } c_{j,i} \tag{19}$$

where *i and j* = 0~6, A_i and B_j are fuzzy number, and $c_{j,i}$ is a real number. The graph of the fuzzy rule table and the fuzzification are shown in Fig. 5.

d. Construct the output of the fuzzy system $u_f(e,de)$ by using the singleton fuzzifier, product-inference rule, and central average defuzzifier method. Although there are total 49 fuzzy rules in Fig. 5 will be inferred, actually only 4 fuzzy rules can be effectively excited to generate a non-zero output. Therefore, if an error e is located between e_i and e_{i+1}, and an error change de is located between de_j and de_{j+1}, only four linguistic values A_i, A_{i+1}, B_j, B_{j+1} and corresponding consequent values $c_{j,i}$, $c_{j+1,i}$, $c_{j,i+1}$, $c_{j+1,i+1}$ can be excited, and the output of the fuzzy system can be inferred by the following expression:

$$u_f(e,de)=\frac{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}c_{m,n}[\mu_{A_n}(e)*\mu_{B_m}(de)]}{\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}\mu_{A_n}(e)*\mu_{B_m}(de)}\triangleq\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}c_{m,n}*d_{n,m} \tag{20}$$

where $d_{n,m}\triangleq \mu_{A_n}(e)*\mu_{B_m}(de)$. And those $c_{i,j}$ are adjustable parameters. In addition, by using (18), it is straightforward to obtain $\sum_{n=i}^{i+1}\sum_{m=j}^{j+1}d_{n,m}=1$ in (20).

2. Reference model (RM):

Second order system is usually as the RM in the adaptive control system. Therefore, the transfer function of the RM in Fig.3 can be expressed as

$$\frac{\psi_m(s)}{\psi_p^*(s)}=\frac{\omega_n^2}{s^2+2\varsigma\omega_n s+\omega_n^2} \tag{21}$$

where ω_n is natural frequency and ς is damping ratio. Furthermore, because the characteristics of no overshoot, fast response and zero steady-state error are the important factors in the design of a PMLSM servo system; therefore, it can be considered as the selective criterion of ω_n and ς. The design methodology is described as follows: Firstly, the (21) matches the requirement of a zero steady-state error condition. Secondly, if we choose $\varsigma\geq 1$, it can guarantee no overshoot condition. Especially, the critical damp value $\varsigma=1$ has

a fastest step response. Hence, the relation between the rising time t_r and the natural frequency ω_n for a step input response in (21) can be derived and shown as follows.

$$(1+\omega_n t_r)e^{-\omega_n t_r} = 0.1 \tag{22}$$

Once the t_r is chosen, the natural frequency ω_n can be obtained. Furthermore, applying the bilinear transformation, (21) can be transformed to a discrete model by

$$\frac{\psi_m(z^{-1})}{\psi_p^*(z^{-1})} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \tag{23}$$

and the difference equation is written as.

$$\psi_m(k) = -b_1\psi_m(k-1) - b_2\psi_m(k-2) + a_0\psi_p^*(k) + a_1\psi_p^*(k-1) + a_2\psi_p^*(k-2) \tag{24}$$

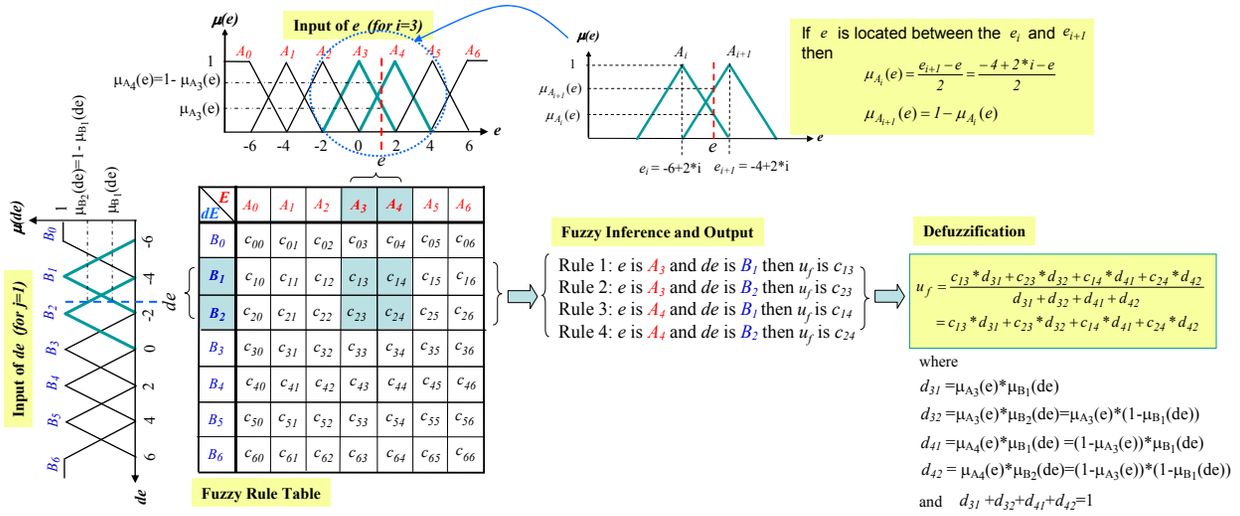

Figure 5. The symmetrical triangular membership function of *e* and *de*, fuzzy rule table, fuzzy inference and fuzzification

3. Parameter adjusting mechanism:

The gradient descent method is used to derive the AFC control law in Fig. 3. The objective of the parameters adjustment in FC is to minimize the square error between the mover position and the output of the RM. The instantaneous cost function is defined by

$$J(k+1) = \frac{1}{2}e_m(k+1)^2 = \frac{1}{2}\left[\psi_m(k+1) - \psi_p(k+1)\right]^2 \tag{25}$$

and the four defuzzifier parameters of $c_{j,i}$, $c_{j+1,i}$, $c_{j,i+1}$, $c_{j+1,i+1}$ are adjusted according to

$$\Delta c_{m,n}(k+1) \propto -\frac{\partial J(k+1)}{\partial c_{m,n}(k)} = -\alpha\frac{\partial J(k+1)}{\partial c_{m,n}(k)} \tag{26}$$

with $m = j, j+1$, $n = i,i+1$ and where α represents learning rate. However, following the similar derivation with (Kung & Tsai, 2007), the $\Delta c_{m,n}$ can be obtained as

$$\Delta c_{m,n}(k) \approx \alpha(K_p + K_i)K_v e(k)d_{n,m} \tag{27}$$

with $m = j, j+1$ and $n = i,i+1$.

2.3. Motion trajectory planning of X-Y table

The circular, window and star motion trajectories are typical used as the performance evaluation of the motion controller for X-Y table.

a. In circular motion trajectory, it is computed by

$$x_i = r \sin(\theta_i) \tag{28}$$

$$y_i = r \cos(\theta_i) \tag{29}$$

with $\theta_i = \theta_{i-1} + \Delta\theta$. Where $\Delta\theta$, r, x_i, y_i are angle increment, radius, X-axis trajectory command and Y-axis trajectory command, respectively.

b. The window motion trajectory is shown in Fig.6. The formulation is derived as follows:

a-trajectory:

$$x_i = x_{i-1}, y_i = S + y_{i-1} \tag{30}$$

b-trajectory:

$$(\theta_i : \frac{6}{4}\pi \to 2\pi, \text{ and } \theta_i = \theta_{i-1} + \Delta\theta)$$

$$x_i = O_{x1} + r\cos(\theta_i), y_i = O_{y1} + r\sin(\theta_i) \tag{31}$$

c-trajectory:

$$x_i = S + x_{i-1}, y_i = y_{i-1} \tag{32}$$

d-trajectory:

$$(\theta_i : \pi \to \frac{6}{4}\pi, \text{ and } \theta_i = \theta_{i-1} + \Delta\theta)$$

$$x_i = O_{x2} + r\cos(\theta_i), y_i = O_{y2} + r\sin(\theta_i) \tag{33}$$

e-trajectory:

$$x_i = x_{i-1}, y_i = -S + y_{i-1} \tag{34}$$

f-trajectory:

$$(\theta_i : \frac{1}{2}\pi \to \pi, \text{ and } \theta_i = \theta_{i-1} + \Delta\theta)$$

$$x_i = O_{x3} + r\cos(\theta_i), y_i = O_{y3} + r\sin(\theta_i) \tag{35}$$

g-trajectory:

$$x_i = -S + x_{i-1}, y_i = y_{i-1} \tag{36}$$

h-trajectory:

$$(\theta_i : 0 \to \frac{1}{2}\pi, \text{ and } \theta_i = \theta_{i-1} + \Delta\theta)$$

$$x_i = O_{x4} + r\cos(\theta_i), y_i = O_{y4} + r\sin(\theta_i) \tag{37}$$

i-trajectory:

$$x_i = x_{i-1}, y_i = S + y_{i-1} \tag{38}$$

where S, $\Delta\theta$, x_i, y_i are position increment, angle increment, X-axis trajectory command and Y-axis trajectory command, respectively. In addition, the (O_{x1}, O_{y1}), (O_{x2}, O_{y2}), (O_{x3}, O_{y3}), (O_{x4}, O_{y4}) are arc center of b-, d-, f-, and h-trajectory in the Fig. 6 and r is the radius. The motion speed of the table is determined by $\Delta\theta$.

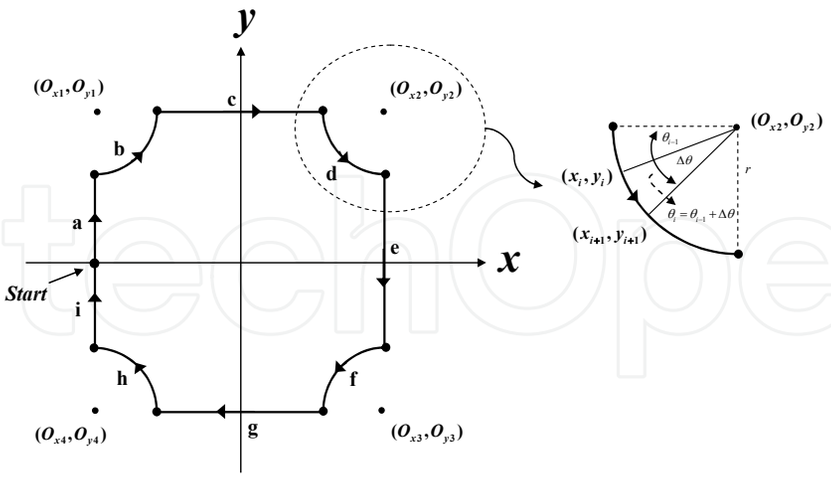

Figure 6. Window motion trajectory

c. Star motion trajectory is shown in Fig.7. The formulation is derived as follows:

a-trajectory :

$$x_i = S + x_{i-1}, \qquad y_i = y_{i-1} \tag{39}$$

b-trajectory :

$$x_i = -S * \sin 54^o + x_{i-1}, \qquad y_i = -S * \sin 36^o + y_{i-1} \tag{40}$$

c-trajectory :

$$x_i = S * \sin 18^o + x_{i-1}, \qquad y_i = S * \sin 72^o + y_{i-1} \tag{41}$$

d-trajectory :

$$x_i = S * \sin 18^o + x_{i-1}, \qquad y_i = -S * \sin 72^o + y_{i-1} \tag{42}$$

e-trajectory :

$$x_i = -S * \sin 54^o + x_{i-1}, \qquad y_i = S * \sin 36^o + y_{i-1} \tag{43}$$

Where S , x_i , y_i are position increment, X-axis trajectory command and Y-axis trajectory command, respectively. The motion speed of the table is determined by S .

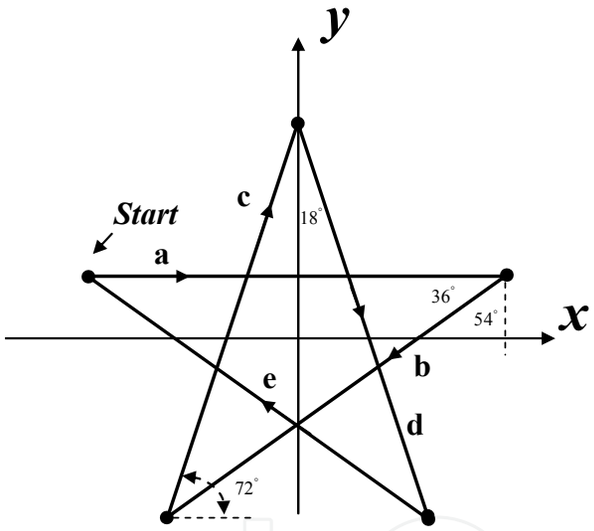

Figure 7. Star motion trajectory

3. The design of a motion control IC for linear motor drive X-Y table

Figure 8 illustrates the internal architecture of the proposed FPGA-based motion control IC for linear motor drive X-Y table. The FPGA uses Altera Stratix II EP2S60, which has 48,352 ALUTs (Adaptive Look-Up Tables), 36 DSP blocks, 144 embedded multipliers, 718 maximum user I/O pins, total 2,544,192 RAM bits, and a Nios II embedded processor which has a 32-bit configurable CPU core, 16 M byte flash memory, 1 M byte SRAM and 16 M byte SDRAM. The Nios II processor can be downloaded into FPGA to construct a SoPC environment. The internal circuit in Fig. 8 comprises a Nios II embedded processor IP (Intellectual Properties) and an application IP. The Nios II processor is depicted to both generate the motion trajectory and collect the response data. The application IP includes the

circuits of two position AFC and speed P controllers as well as two current vector controllers for X-axis and Y-axis table. The sampling frequency of position control loop is designed with 2kHz. The operating clock rate of the designed FPGA controller is 50MHz and the frequency divider generates 50 Mhz (*Clk*), 25 MHz (*Clk-step*), 2 kHz (*Clk-sp*) and 16 kHz (*Clk-cur*) clock to supply all module circuits of application IP in Fig. 8.

An FSM is also employed to model the AFC of the position loop and P controller of the speed loop in X-axis table and shown in Fig. 9, which uses one adder, one multiplier, a look-up table, comparators, registers, etc. and manipulates 35 steps machine to carry out the overall computation. With exception of the data type in reference model are 24-bits, others data type are designed with 12-bits length, 2's complement and Q11 format. Although the algorithm of AFC is highly complexity, the FSM can give a very adequate modeling and easily be described by VHDL. Furthermore, steps s_0~s_6 execute the computation of reference model output; steps s_6~s_9 are for the computation of mover velocity, position error and error change; steps s_9~s_{12} execute the function of the fuzzification; s_{13} describe the look-up table and s_{14}~s_{22} defuzzification; and steps s_{23}~s_{34} execute the computation of velocity and current command output, and the tuning of fuzzy rule parameters. The SD is the section determination of *e* and *de* and the RS,1 represents the right shift function with one bit. The operation of each step in Fig.9 can be completed within 40ns (25 MHz clock) in FPGA; therefore total 35 steps need a 1.4μs operation time. It doesn't loss any control performance for the overall system because the operation time with 1.4μs is much less than the sampling interval, 500 μs (2 kHz), of the position control loop in Fig.3. In Fig. 8, the QEP circuit and circuit for current vector control refer to (Kung & Tsai, 2007). Further, the Nios II embedded processor IP is depicted to perform the function of the motion trajectory and two-axis position/speed loop controller for X-Y table in software. Figure 10 illustrates the flow charts of the main program and the interrupt service routine (ISR), where the interrupt interval is designed with 2ms. All programs are coded in the C programming language. Then, through the complier and linker operation in the Nios II IDE (Integrated Development Environment), the execution code is produced and can be downloaded to the external Flash or SDRAM via JTAG interface.

Under the proposed design method, the overall resource usage of the proposed motion control IC is listed in Table 1 which the two AFC circuits need 16,110 ALUTs, the Nios II embedded processor IP needs 8,275 ALUTs and 46,848 RAM bits and the application IP needs 22,928 ALUTs and 595,968 RAM bits in FPGA. Therefore, the motion control IC uses 64.5% ALUTs resource and 25.2% RAM resource of Stratix II EP2S60.

IP	Module circuit	ALUTs	Memory (bits)
Nios II Embedded Processor IP		8,275	46,848
Application IP	2 x Adaptive fuzzy controller (AFC)	16,110	0
	2 x Current loop controller (Current vector control, SVPWM,ADC,QEP)	6,818	595,968
Total		31,203	642,816

Table 1. The resource usage of a motion control IC in FPGA

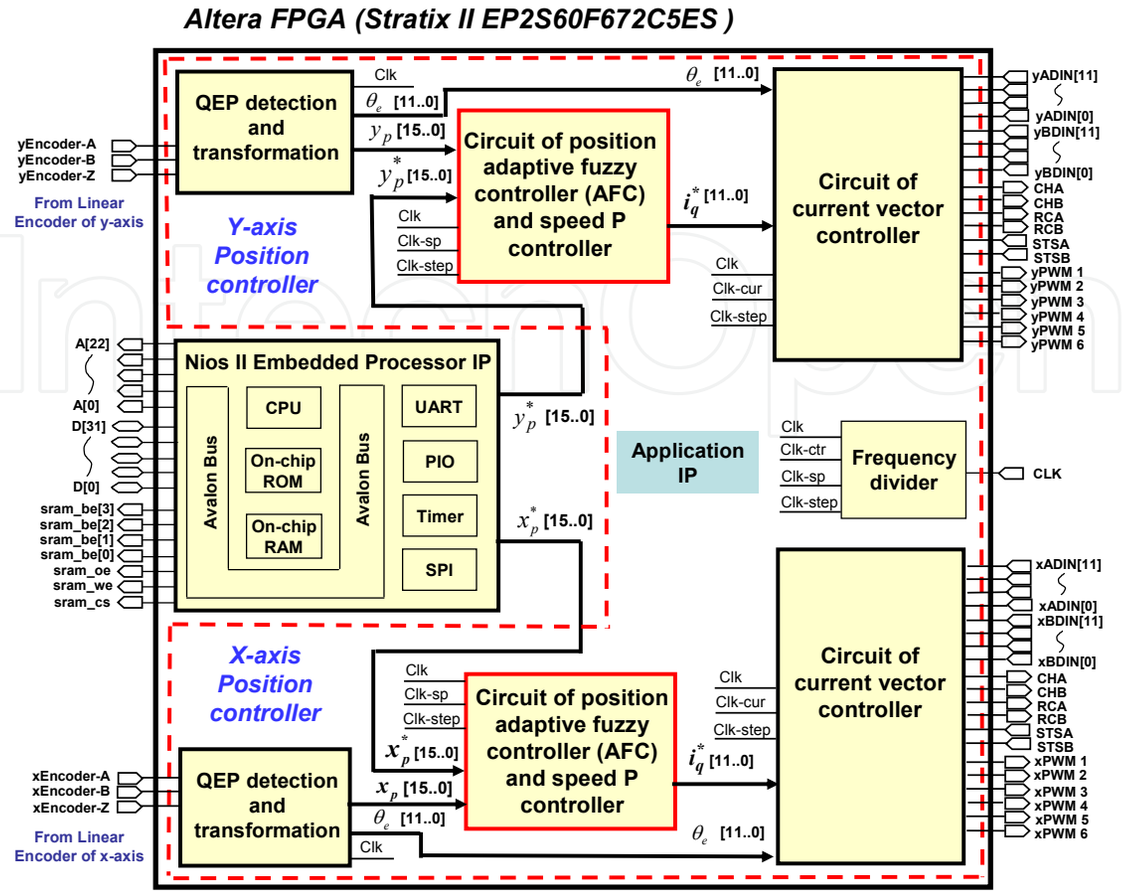

Figure 8. The internal architecture of a motion control IC for linear motor drive X-Y table

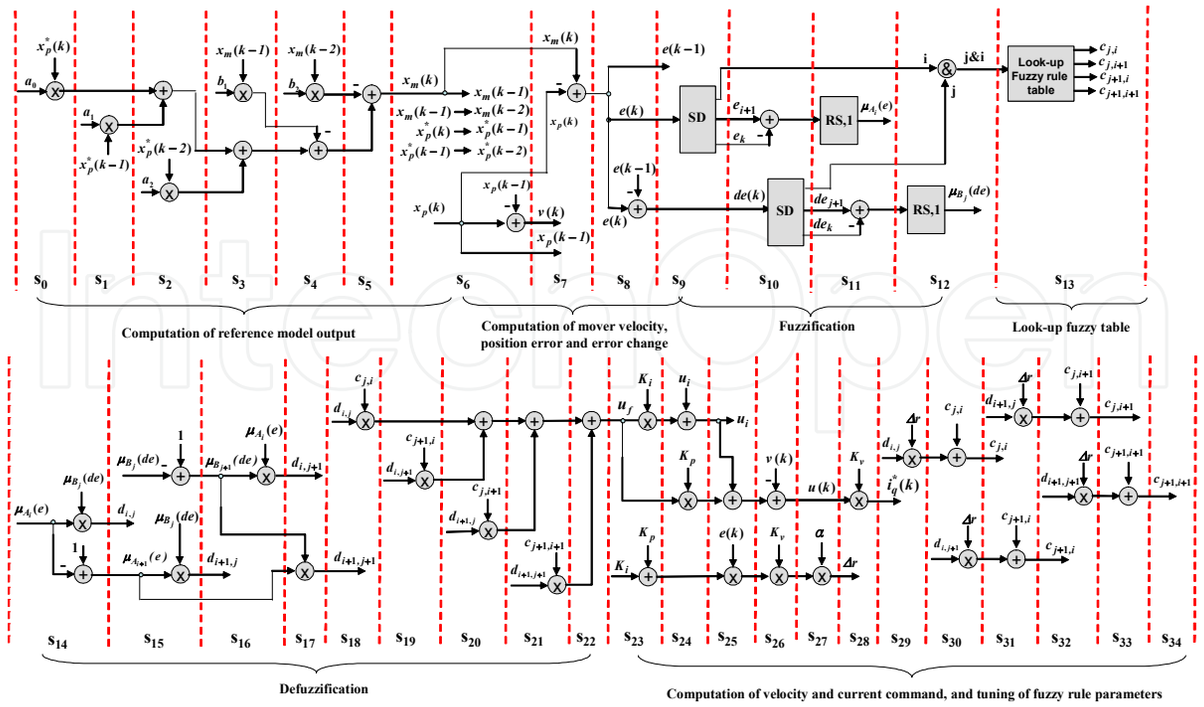

Figure 9. State diagram of an FSM for describing the AFC in position loop and P controller in speed loop (for X-axis Table)

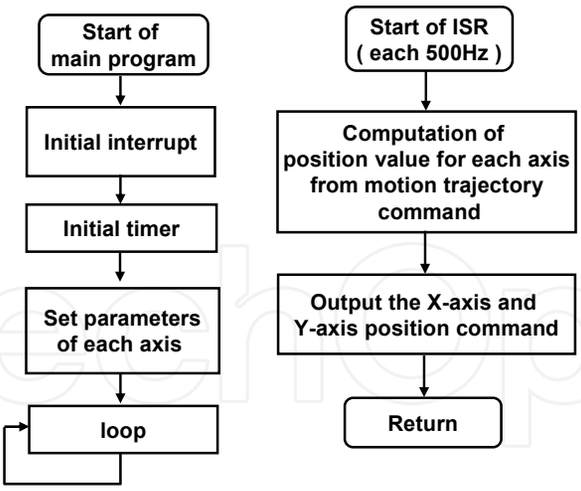

Figure 10. Flow chart of the main and ISR program in Nios II embedded processor

4. Experimental results

The overall experimental system depicted in Fig.3 includes an FPGA (Stratix II EP2S60F672C5) experimental board, two voltage source IGBT inverters and an X-Y table which is driven by two PMLSMs. The PMLSM was manufactured by the BALDOR electric company; and it is a single-axis stage with a cog-free linear motor and a stroke length with 600mm. The parameters of the motor are: $R_s = 27\,\Omega$, $L_d = L_q = 23.3$ mH, $K_t = 79.9$N/A. The input voltage, continuous current, peak current (10% duty) and continuous power of the PMLSM are 220V, 1.6A, 4.8A and 54W, respectively. The maximum speed and acceleration are 4m/s and 4 g but depend on external load. The moving mass is 2.5Kg, the maximum payload is 22.5Kg and the maximum thrust force is 73N under continuous operating conditions. A linear encoder with a resolution of 5μm is mounted on the PMLSM as the position sensor, and the pole pitch is 30.5mm (about 6100 pulses). The inverter has three sets of IGBT power transistors. The collector-emitter voltage of the IGBT is rated 600V; the gate-emitter voltage is rated ±20V, and the DC collector current is rated 25A and in short time (1ms) is 50A. The photo-IC, Toshiba TLP250, is used in the gate driving circuit of IGBT. Input signals of the inverter are PWM signals from the FPGA device.

To confirm the effectiveness of the proposed AFC in linear drive X-Y table, a realization of position controller based on the FPGA in Fig.3 is constructed and some experiments are evaluated. The control sampling frequency of the current, speed and position loops are designed as 16kHz, 2kHz and 2kHz, respectively. In the motion control IC, two position/speed/current controllers are all realized by hardware in FPGA, and the motion trajectory algorithm is implemented by software using the Nios II embedded processor. The speed controller adopts a P controller and the AFC is used in the position loop. The transfer function of the reference model is selected by a second order system with the natural frequency of 30 rad/s and damping ratio of 1. The step response is first tested to evaluate the performance of the proposed controller. Figures 11 and 12 respectively show the position step responses for X-axis and Y-axis table using the FC (learning rate=0) and AFC (learning

rate=0.1). The position command is a 4/3Hz square wave signal with 10mm amplitude. In Figs. 11(a) and 12(a), when an 11 kg load is added upon the mover of the X-Y table and the fuzzy control by using a fixed rule table, the position dynamic response in X-axis and Y-axis table exhibits a 12.8% and 23.1% overshoot and severe oscillation, respectively. Accordingly, an AFC is adopted in Fig.3. When the proposed AFC is used with learning rate being 0.1, the tracking results are highly improved and presented in Figs. 11(c) and 12(c). Initially, the position response in X-axis or Y-axis table tracks the output of the reference model with oscillation. After one or two square wave commands, the $c_{i,j}$ parameters in fuzzy rule table are tuned to adequate values, and the position response in X-axis or Y-axis table can closely follow the output of the reference model. Further, the tracking motion about circular, window and star trajectory by using FC and AFC are experimented. To evaluate the tracking performance, the indices are firstly defined as follows.

$$T(k) = \sqrt{(x_m(k) - x_p(x))^2 + (y_m(k) - y_p(x))^2} \tag{44}$$

$$m = \sum_{k=1}^{n} T(k) / n \tag{45}$$

$$\sigma = \sqrt{\sum_{k=1}^{n} (T(k) - m)^2 / n} \tag{46}$$

Where $T(k)$, m and σ respectively represent instantaneous value, mean and variance of tracking error. In the circular tracking motion, the circle command is with center (25, 25) cm and radius 10cm and its experimental results are shown in Figs. 13~14. In the window tracking motion, the trajectory is designed as Fig.6 and its experimental results are shown in Figs. 15~16. In the star tracking motion, the trajectory is designed as Fig.7 and its experimental results are shown in Figs. 17~18. Further, the tracking performance in Figs 13~18 by using FC and AFC control algorithm are evaluated according to the indices of (44)~(46), and its results are listed in Table 2. Compared with FC, the mean of tracking errors

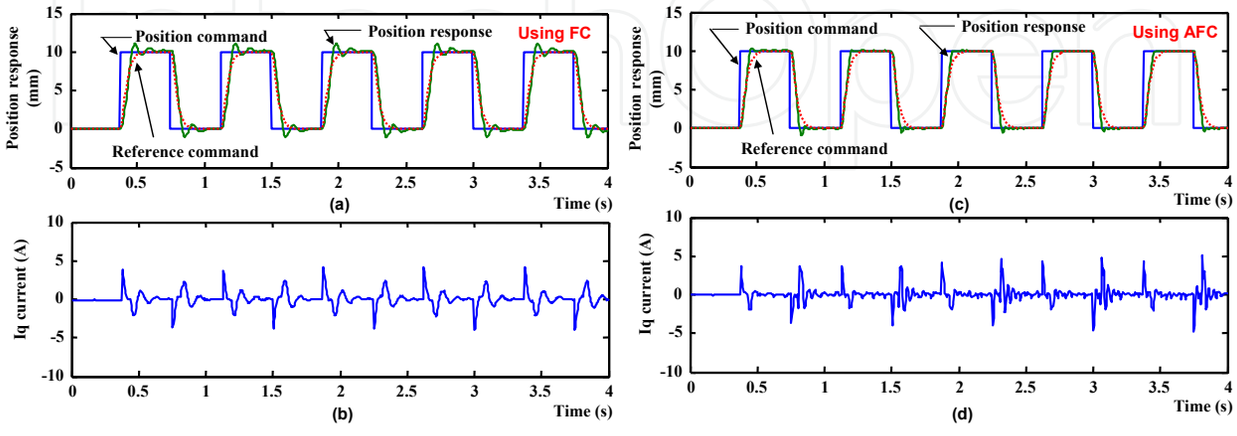

Figure 11. (a) Position step response and (b) current response by using the FC as well as (c) position step response and (d) current response by using the AFC in X-axis table

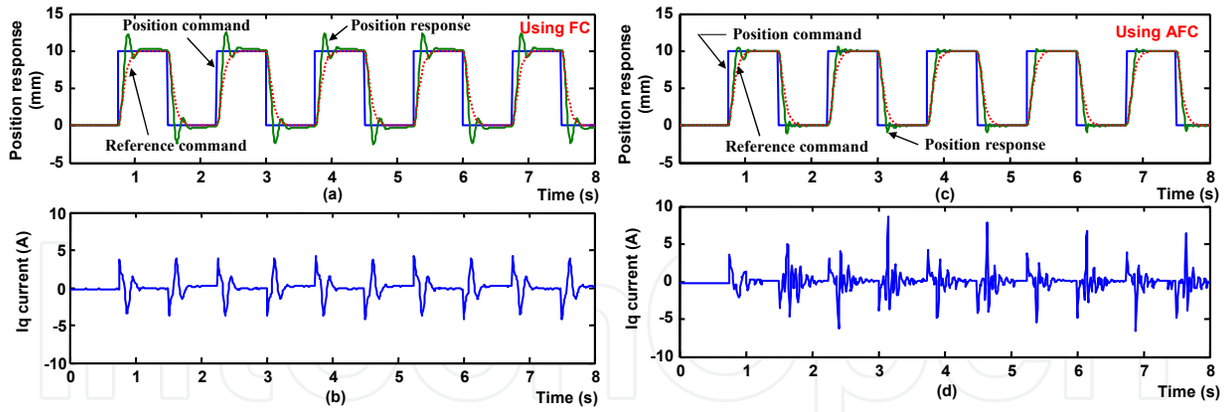

Figure 12. (a) Position step response and (b) current response by using the FC as well as (c) position step response and (d) current response by using the AFC in Y-axis table

in circular, winddow and star motion trajectory are significantly reduced about 41.6%, 14.6% and 12.8% and the variance of tracking errors reduced about 33.3%, 64.6% and 47.4% after using AFC. Therefore, it shows that the AFC has a better tracking performance than FC in motion control of linear motor drive X-Y table. Finally, from the experimental results of Figs.11~18, it demonstrates that the proposed AFC and the FPGA-based motion control IC used for the linear motor drive X-Y table is effective and correct.

5. Conclusion

This study successfully presents a motion control IC for linear motor drive X-Y table based on FPGA technology. The works herein are summarized as follows.

1. The functionalities required to build a fully digital motion controller of linear motor drive X-Y table, such as the two current vector controllers, two speed P controllers, and two position AFCs and one motion trajectory planning, have been integrated in one FPGA chip.
2. An FSM joined by one multiplier, one adder, one LUT, or some comparators and registers has been employed to model the overall AFC algorithm, such that it not only is easily implemented by VHDL but also the resources usage can be reduced in the FPGA.
3. The software/hardware co-design technology under SoPC environment has been successfully applied to the motion controller of linear motor drive X-Y table.

However, the experimental results by step response as well as the circular, window and star motion trajectory tracking, has been revealed that the software/hardware co-design technology with the parallel processing well in the motion control system of linear motor drive X-Y table.

Control algorithm	Fuzzy controller (FC)			Adaptive fuzzy controller (AFC)		
Tracking error (mm)	Circular motion	Window motion	Star motion	Circular motion	Window motion	Star motion
Mean	3.51	1.43	1.64	2.05	1.22	1.43
Variance	0.60	2.12	3.52	0.40	0.75	1.85

Table 2. Evaluation of tracking performance using FC and AFC

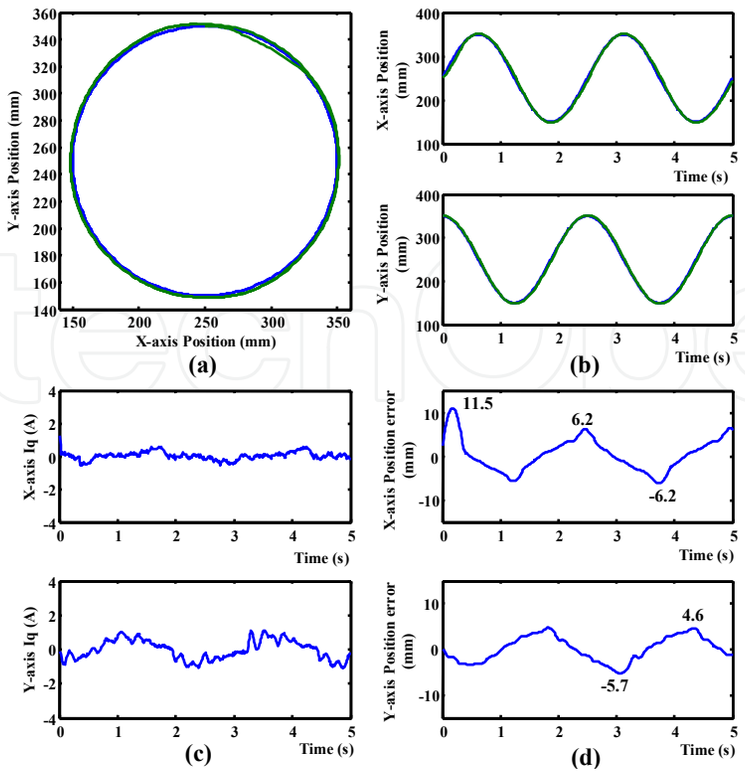

Figure 13. Circular trajectory response by using the FC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

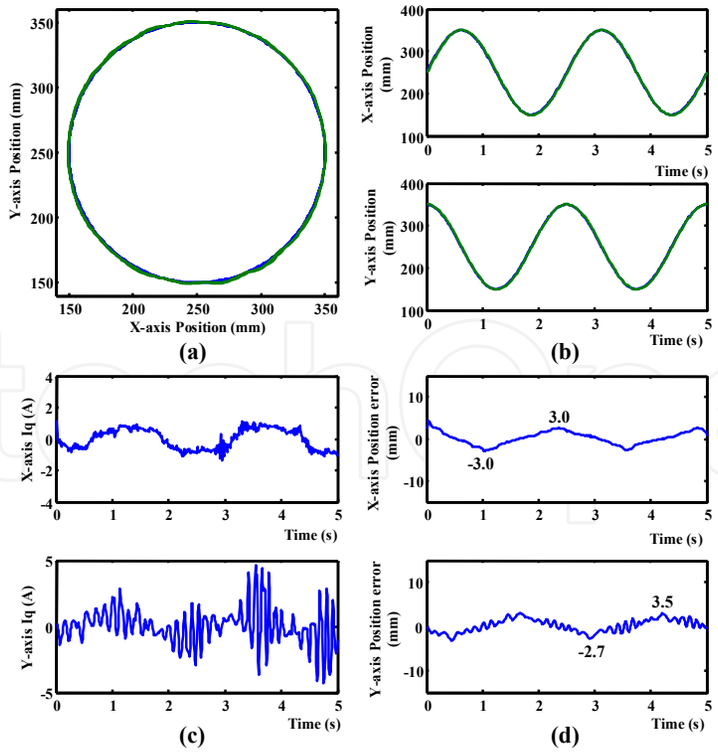

Figure 14. Circular trajectory response by using the AFC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

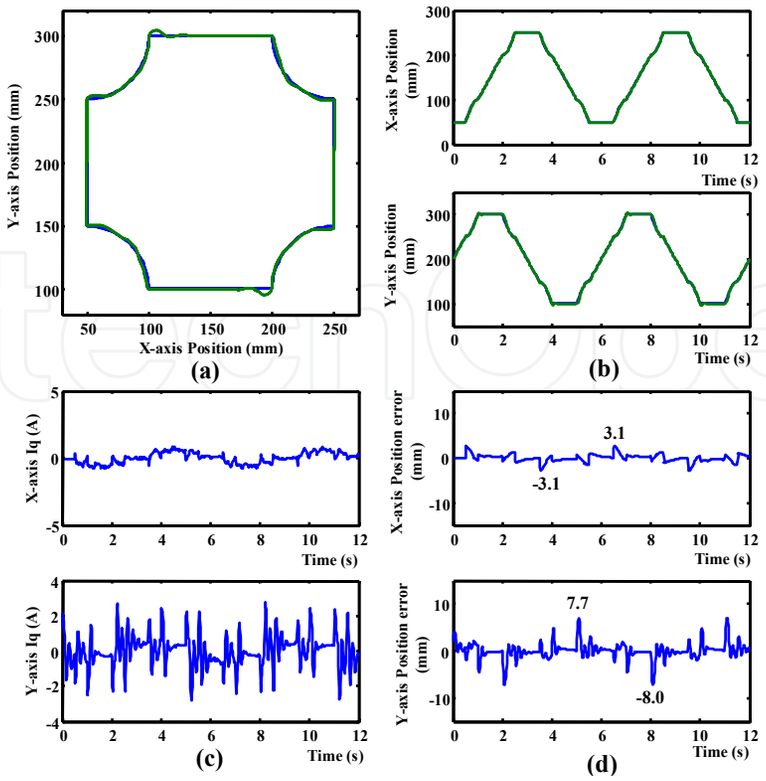

Figure 15. Window trajectory response by using the FC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

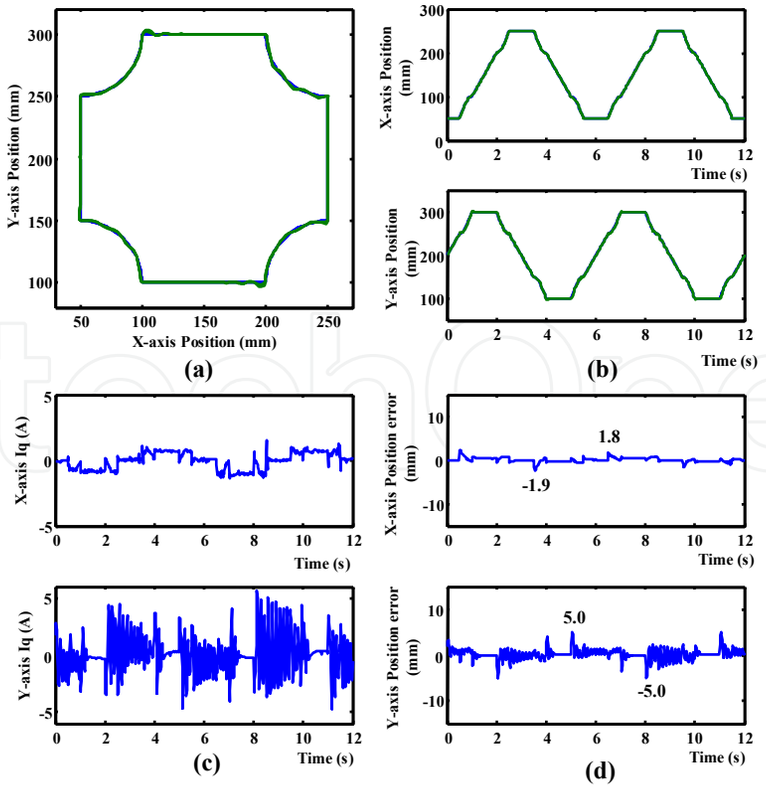

Figure 16. Window trajectory response by using the AFC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

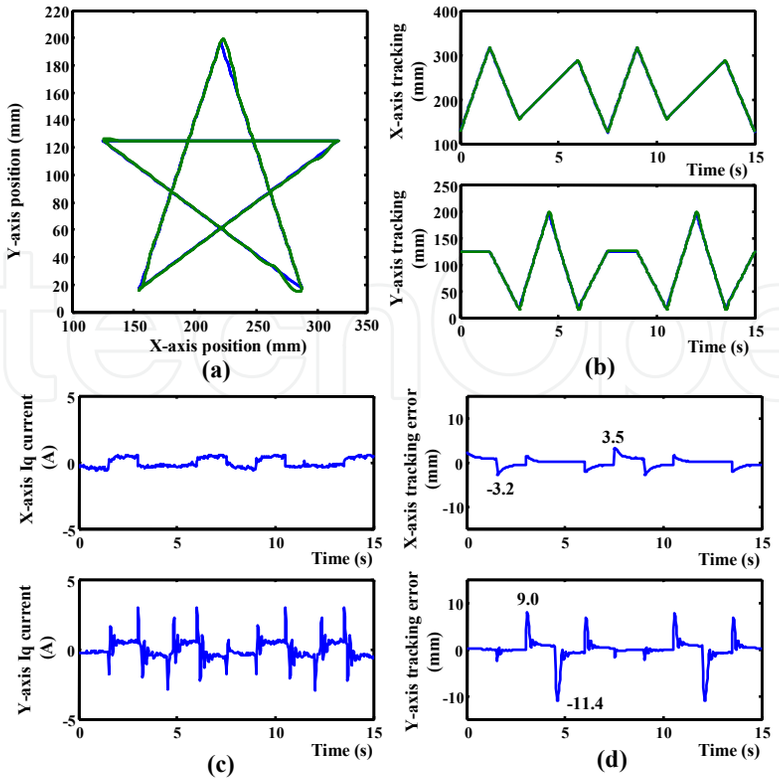

Figure 17. Star trajectory response by using the FC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

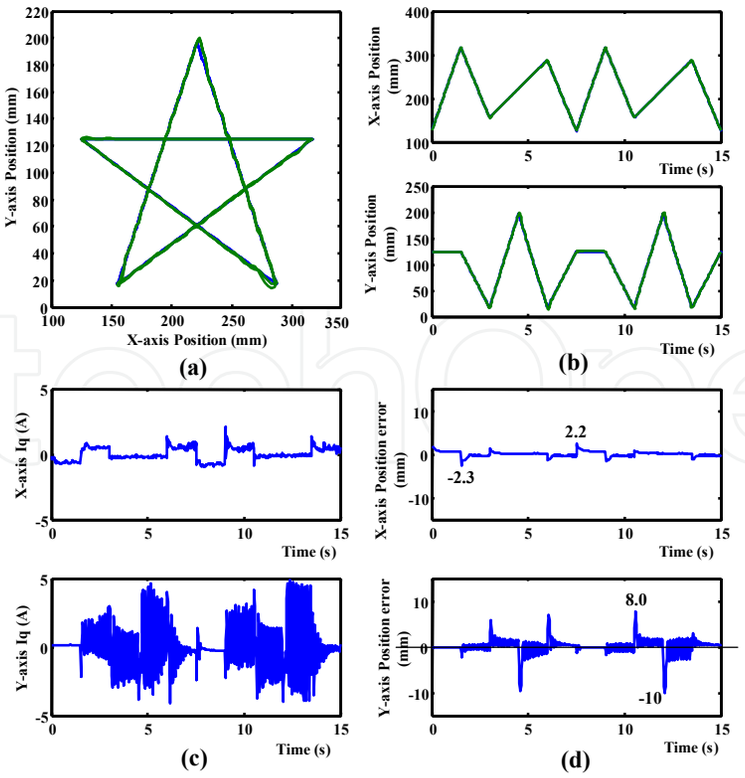

Figure 18. Star trajectory response by using the AFC (a) Star trajectory tracking (b) Position tracking in X- and Y-axis table (c) Control efforts in X- and Y-axis table (d) Tracking errors in X- and Y-axis table

Author details

Ying-Shieh Kung
Department of Electrical Engineering, Southern Taiwan University, Taiwan

Chung-Chun Huang and Liang-Chiao Huang
Green Energy and Environment Research Laboratories, Industrial Technology Research Institute, Taiwan

Acknowledgement

The financial support provided by Bureau of Energy is gratefully acknowledged.

6. References

Cho, J. U., Le, Q. N. and Jeon, J. W. (2009) An FPGA-based multiple-axis motion control chip, *IEEE Trans. Ind. Electron.*, vol. 56, no.3, pp.856-870.

Groove, M.P. (1996) *Fundamentals of modern manufacturing: materials, process, and systems*. Prentice Hall, Upper Saddle River, NJ.

Goto, S., Nakamura, M. and Kyura, N. (1996) Accurate contour control of mechatronic servo systems using gaussian networks. *IEEE Trans. Ind. Electron.*, vol.43, no. 4, pp. 469-476.

Hanafi, D., Tordon, M. and Katupitiya, J. (2003) An active axis control system for a conventional CNC machine. *Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 1188-1193.

Hall, T.S., Hamblen, J.O. (2004) System-on-a-programmable -chip development platforms in the classroom. *IEEE Trans. on Education*, vol.47, no.4, pp.502-507.

Lin, F.J., SHieh, P.H. and Shen, P.H. (2006) Robust recurrent-neural-network sliding-mode control for the X-Y table of a CNC machine. *IEE Proc. Control Theory Application*, vol. 153, no. 1, pp. 111-123.

Monmasson, E., Idkhajine L. and Naouar, M.W. (2011) FPGA-based Controllers. *IEEE Trans. Electron. Magaz.*, vol. 5, no.1, pp.14-26.

Kung, Y.S., Huang P.G. and Chen, C.W. (2004) Development of a SOPC for PMSM drives. *Proceedings of the IEEE International Midwest Symposium on Circuits and Systems*, vol. II, pp. II-329~II-332.

Kung, Y.S. and Tsai, M.H. (2007) FPGA-based speed control IC for PMSM drive with adaptive fuzzy control. *IEEE Trans. on Power Electronics*, vol. 22, no. 6, pp. 2476-2486.

Kung, Y.S. and Chen, C.S. (2008). Realization of a motion control IC for robot manipulator based on novel FPGA technology. *Robot manipulators, programming, design and control.* pp.291~312, I-Tech, Vienna.

Sanchez-Solano, S., Cabrera, A. J., Baturone, I., Moreno-Velo, F.J., Brox, M. (2007) FPGA implementation of embedded fuzzy controllers for robotic applications. *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1937-1945.

SOPC World, (2004) Altera Corporation.

Wang, G.J. and Lee, T.J. (1999) Neural-network cross-coupled control system with application on circular tracking of linear motor X-Y table. *International Joint Conference on Neural Networks*, pp. 2194-2199.