

# FPGA-Implementation of Inverse Kinematics and Servo Controller for Robot Manipulator

Ying-Shieh Kung, *IEEE Member*, Kuan-Hsuan Tseng  
and Chia-Sheng Chen

*Department of Electrical Engineering  
Southern Taiwan University of Technology  
Yung-Kang, Tainan County, 710, TAIWAN  
kung@mail.stut.edu.tw*

Hau-Zen Sze and An-Peng Wang

*Energy & Resources Laboratories  
Industrial Technology Research Institute  
Chutung, Hsinchu County, 310, Taiwan  
HauZenSze@itri.org.tw*

**Abstract** - The implementation of inverse kinematics and servo controller for robot manipulator using FPGA (Field Programmable Gate Array) is investigated in this paper. Firstly, the mathematical model and the servo controller of robot manipulator are described. Secondly, the inverse kinematics is formulated. Thirdly, the circuit design to implement the function of inverse kinematics and servo controller based on FPGA is introduced. To reduce the usage of the logic elements (LEs) in FPGA, a finite state machine (FSM) method by overall 42 steps to implement the computation of inverse kinematics is applied. Finally, to evaluate the performance of the proposed design circuits, an experimental system consisted by the proposed FPGA-based motion controller and a Mitsubishi RV-M1 micro robot are set up and some experimental results are demonstrated the correctness and effectiveness.

**Index Terms** - *FPGA, Robot manipulator, Finite state machine, Inverse kinematics.*

## I. INTRODUCTION

Robotic control is currently an exciting and highly challenging research focus. Several solutions for implementing the control architecture for robots have been proposed [1-4]. M. Kabuka et al [1], applied two high-performance floating-point signal processors and a set of dedicated motion controllers to build a control system for a six-joint robots arm. G. Yasuda [2] adopted a PC-based microcomputer and several PIC microcomputers to construct a distributed motion controller for mobile robots. T.S. Li et al [3], utilized an FPGA to implement autonomous fuzzy behaviour control on mobile robot. S.N. Oh et al [4] presented a DSP and an FPGA to design the overall hardware system in controlling the motion of biped robots. Y.S. Kung et al [5] employed hardware/software co-design technology applying to the motion control of an articulated robot arm. However, these methods did not provide the solution for the computation of the complicated formulation in FPGA, such as inverse kinematics, and they also did not consider how to reduce the resource usage in FPGA but without loss the system performance.

Due to the advantages of their programmable hardware feature, short-to-market, high speed, and higher density for the implementation, FPGA has brought more attention and become a popular research topic on digital

control [6-9]. In this paper, the FPGA-implementation of inverse kinematics and servo controller for robot manipulator is presented. The FPGA is a kernel of the proposed robot system. The function of inverse kinematics, five-axis position controller circuits, five axis speed controller circuits, five sets of the PWM generation circuits and five sets of QEP detection circuits are all implemented in an FPGA chip. The FPGA chip herein adopts Altera Stratix II EP2S60 [10-11], which has 24,176 ALMs (equivalent 60,440LEs), maximum 492 user I/O pins, 36 DSP blocks, 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.

## II. SYSTEM DESCRIPTION AND HARDWARE DESIGN

The detailed theoretical model of robot manipulator and the design methodology for the proposed FPGA-based motion controller are described below.

### A. The mathematical model of robotic manipulator with actuator

The dynamic equation of the n-link robot manipulator is given by: [12]

$$M(q)\ddot{q} + V_m(q, \dot{q})\dot{q} + F(\dot{q}) + G(q) = \tau \quad (1)$$

with  $M(q)$  inertial matrix,  $V_m(q, \dot{q})$  Coriolis /centripetal vector,  $G(q)$  gravity vector,  $F(\dot{q})$  friction vector. The  $q$ ,  $\dot{q}$ ,  $\ddot{q}$  and  $\tau$  denote the n-vector of the position, velocity, acceleration and generalized forces, respectively. The  $\tau$ ,  $q$ ,  $\dot{q}$  and  $\ddot{q}$  are  $R^n$ . The dynamics of the dc motors that drive the arm links are given by the  $n$  decoupled equations [12]

$$J_M \ddot{q}_M + B \dot{q}_M + F_M + R\tau = K_M v \quad (2)$$

with

$$\begin{aligned} q_M &= \text{vec}\{q_{M_i}\}, J_M = \text{diag}\{J_{M_i}\} \\ B &= \text{diag}\{B_{M_i} + K_{b_i} K_{M_i} / R_{a_i}\} \triangleq \text{diag}\{B_i\} \\ R &= \text{diag}\{R_i\}, K_M = \text{diag}\{K_{M_i} / R_{a_i}\} \\ v &= \text{vce}(v_i), \tau = \text{vec}(\tau_i) \end{aligned} \quad (3)$$

where  $R_{a_i}$ ,  $i_i$ ,  $v_i$ ,  $K_{b_i}$ ,  $K_{M_i}$  are resistance, current, voltage, voltage constant, current constant, respectively. The  $J_{M_i}$ ,

$B_{M_i}$ ,  $\tau_{M_i}$ ,  $\tau_{L_i}$  are the inertial, viscous, generated torque of the motor, load torque for the  $i^{\text{th}}$  axis of the robot arm, respectively. Combining (1) and (2), the dynamic equations of robot arm and dc motor can be obtained. At first, if the gear ratio of the coupling from the  $i^{\text{th}}$  motor to the  $i^{\text{th}}$  arm link is  $r_i$ , we can define as follows

$$q_i = r_i q_{M_i} \text{ or } q = R q_M \quad (4)$$

Substituting (4) into (2), then into (1), it can be obtained

$$(J_M + R^2 M(q))\ddot{q} + (B + R^2 V_m(q, \dot{q}))\dot{q} + R F_M(q) + R^2 F(q) + R^2 G(q) = R K_M v \quad (5)$$

In commercial robot arm, to increase torque output, the gear ratio  $r_i$  is often chosen by small value. So, it can help us to simply the formulations. From (5), the dynamic equation combined the motor and arm link is given by

$$(J_{M_i} + r_i^2 m_{ii})\ddot{q}_i + B_i \dot{q}_i + r_i F_{M_i} = \frac{r_i K_{M_i}}{R_{a_i}} v_i - r_i^2 d_i \quad (6)$$

where

$$d_i = \sum_{j \neq i} m_{ij} \ddot{q}_j + \sum_{j,k} V_{jki} \dot{q}_j \dot{q}_k + F_i + G_i \quad (7)$$

Therefore, when the gear ratio  $r_i$  is small, the  $r_i^2$  term in (6) can be neglected, and it is simplified by

$$J_{M_i} \ddot{q}_i + B_i \dot{q}_i + r_i F_{M_i} = \frac{r_i K_{M_i}}{R_{a_i}} v_i \quad (8)$$

Substituting (3) and (4) into (8), the following equations can be given

$$J_{M_i} \ddot{q}_{M_i} + (B_{M_i} + \frac{K_{M_i} K_{b_i}}{R_{a_i}})\dot{q}_{M_i} + T_{L_i} = \frac{K_{M_i}}{R_{a_i}} v_i \quad (9)$$

where  $T_{L_i}$  is the external force or the disturbance force induced by other arm motion. Therefore, if the gear ratio is a small value, the control block diagram combined by arm link, motor actuator, P controller in position and PI controller in speed loop at each axis can be represented in Fig.1. For example, the digital PI controllers of the speed loop in Fig.1 are formulated as follows.

$$v_{p\_j}(n) = k_{p\_j} e_v(n) \quad (10)$$

$$v_{i\_j}(n) = u_{i\_j}(n-1) + k_{i\_j} e_v(n-1) \quad (11)$$

$$v_j(n) = v_{p\_j}(n) + v_{i\_j}(n) \quad (12)$$

Where the  $k_{p\_j}$ ,  $k_{i\_j}$  are P-controller gain and I-controller gain at j-axis, respectively. The  $v_j(n)$  is the PI controller output.

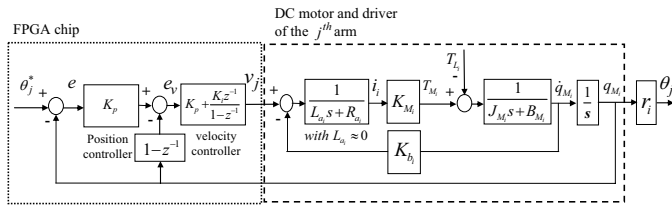


Fig.1 The block diagram of servo position controller for  $j^{\text{th}}$  robot arm

## B. Computational of the Inverse kinematics for robot manipulator

Figure 2 shows the link coordinate system of the five-axis articulated robot manipulator using the Denavit-Hartenberg convention. Table I illustrates the values of the kinematics parameters. The inverse kinematics of the articulated robot is described in detail by author of [13]. The inverse kinematics transform the coordinates of robot manipulator from Cartesian space  $R^3$  ( $x, y, z$ ) to the joint space  $R^5$  ( $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ ) by the following computational procedure.

$$\text{Step1: } \theta_1 = \theta_5 = a \tan 2[y, x] \quad (13)$$

$$\text{Step2: } b = \pm \sqrt{(x^2 + y^2)} \quad (14)$$

$$\text{Step3: } \theta_3 = \cos^{-1} \left( \frac{b^2 + (d_1 - d_5 - z)^2 - a_2^2 - a_3^2}{2a_2 a_3} \right) \quad (15)$$

$$\text{Step4: } S_2 = (a_2 + a_3 \cos \theta_3)(d_1 - d_5 - z) - a_3 b \sin \theta_3 \quad (16)$$

$$\text{Step5: } C_2 = (a_2 + a_3 \cos \theta_3)b + a_3 \sin \theta_3 (d_1 - d_5 - z) \quad (17)$$

$$\text{Step6: } \theta_2 = a \tan 2[S_2, C_2] \quad (18)$$

$$\text{Step7: } \theta_4 = -\theta_2 - \theta_3 \quad (19)$$

where  $a_1, a_5, d_1, d_5$  are Denavit-Hartenberg parameters and atan2 function refers to TABLE II.

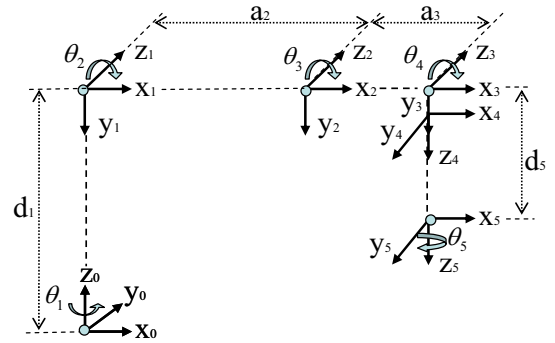


Fig.2 Link coordinates of a five-axis articulated robot

TABLE I  
Denavit-Hartenberg parameters of the robot manipulator

	$\theta$	$d$	$a$	$\alpha$
1	$\theta_1$	$d_1$ (300 mm)	(0 mm)	$\alpha_1(-\pi/2)$
2	$\theta_2$	(0 mm)	$a_2$ (250 mm)	0
3	$\theta_3$	(0 mm)	$a_3$ (160 mm)	0
4	$\theta_4$	(0 mm)	(0 mm)	$\alpha_4(-\pi/2)$
5	$\theta_5$	$d_5$ (72 mm)	(0 mm)	0

TABLE II  
Four-quadrant arctan function [13]

Case	Quadrants	atan2 (y, x)
$x > 0$	1, 4	$\arctan (y / x)$
$x = 0$	1, 4	$[\text{sgn} (y)]\pi/2$
$x < 0$	2, 3	$\arctan (y / x) + [\text{sgn} (y)]\pi$

### C. Circuits design of the inverse kinematics and servo controller for robot manipulator

Figure 3 illustrates the internal architecture of the proposed FPGA-based motion controller for a robot manipulator. In Fig.3, the components have a Nios II processor IP (Intelligent Properties) and our proposed motion controller IP. The motion controller IP includes the circuits of inverse kinematics computation, five axis position controller, five-axis speed controller, five set of PWM generation and five sets of QEP detection. The FPGA chip adopts herein is an Altera Stratix II EP2S60, which has 24,176 ALMs (equivalent 60,440LEs), maximum 492 user I/O pins, 36 DSP blocks, 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. The Nios II processor depicted in Fig. 3 performs the command generation and five axis position read-in of robot manipulator. In Fig.3, the system frequency is designed with 50MHz. In the implementation of inverse kinematics, if we use parallel processing method, it is resource consumption in FPGA; therefore, the finite state machine method is applied. In Fig.4, there are 42 steps to perform the inverse kinematics in (13)~(19), and the circuit needs 3 multipliers, 2 dividers, 2 adders, 1 square root function, 1 component for arctan function, 1 component for arcs function, 1 look-up-table for sin function and some comparator for atan2 function. From the simulation results, to realize the inverse kinematics needs 840ns (20ns/step\* 42 steps) computation time and uses 6,994 ALMs (17,485 LEs) resource. The computation for inverse kinematics takes 5.6ms for a C program by Nios II processor in software [5] and the computation time is about 6,666 times slower than the computation time on FPGA-hardware. Besides, in the implementation of servo controller, the FSM method is also applied to design the controller circuits in position and speed loop for reducing the usages of the logic elements (LEs) in FPGA. The sampling frequency in position loop and speed loop are 762Hz and 1525Hz, respectively. At each sampling time, the step control sequence for computing the controller output for five-axis position loop and five-axis speed loop are arranged in Fig. 5(a) and Fig. 5(b). In position loop, there are 4 steps to compute the P controller at each axis and it is shown in Fig.6. In speed loop, there are 12 steps to estimate the speed value and compute the PI controller at each axis, which is shown in Fig.7. Figure 8 show the PWM circuits. The PWM is designed with 18 kHz frequency value and 1.28 $\mu$ s dead-band. Figure 9 show the QEP circuits. The simulation results of the QEP and PWM circuits are verified but are not shown here. The overall resource usages to implement the motion controller of robot manipulator in Fig.3 are listed in TABLE III.

### III. EXPERIMENTS AND RESULTS

The overall experimental system in this paper includes an FPGA experimental board, five sets of inverter and a Mitsubishi Movemaster RV-M1 micro articulated robot. The micro articulated robot shown in Fig. 10 has five servo axes

(excluding the hand). Each axis is driven by a 24V DC servo motor with a reduction gear. The operation ranges of the articulated robot are wrist roll  $\pm 180$  degrees (J5-axis), wrist pitch  $\pm 90$  degrees (J4-axis), elbow rotation 110 degrees (J3-axis), shoulder rotation 130 degrees (J2-axis) and waist rotation 300 degrees (J1-axis). The gear ratios for J1 to J5 axis of the robot are 1:101, 1:170, 1:114, 1:186 and 1:108, respectively. The maximum path velocity is 1000mm/s and the lifting capacity is 1.2kg including the hand. The total weight of this robot is 19 kg. The FPGA chip adopts Altera Stratix II EP2S60. A Nios II embedded processor can be download to this FPGA chip.

In Fig.3, the realization in PWM switching frequency, dead-band of inverter, position and velocity control sampling frequency are set at 18k Hz, 1.28 $\mu$ s, 762 Hz and 1525 Hz, respectively. Moreover, in the position loop P controller design, the controller parameters at each axis of robot arm are selected with identical values by P-gain with 2.4. However, in the speed loop PI controller design, the controller parameters at each J1~J5 axis are selected with different values by [3.17, 0.05], [3.05, 0.12], [2.68, 0.07], [2.68, 0.12] and [2.44, 0.06]. To confirm the effectiveness of the proposed motion control IC, the square-wave position command with  $\pm 3$  degrees amplitude and 2 seconds period is first adopted to test the dynamic response performance. At the beginning of the step response testing, the robot arm is translated to a specified attitude for joints J1-J5 rotating at the [9°, 40°, 60°, 45°, 10°] position. Figures 11~12 show the experimental results of the step response at J1 and J5 axis under these design conditions, where the rising times of step response are with 109ms and 180 ms for axis J1 and J5. The results also indicate that these step responses have almost zero steady-state error and no oscillation. Next, to test the performance of trajectory tracking of the robot manipulator, a linear trajectory path is specified where the robot manipulator moves from the starting position, (200, 100, 400) mm to the ending position, (300, 0, 300) mm, then back to starting position. The linear trajectory command is generated 100 equidistant segmented points from the starting to the ending position. Each middle position at Cartesian space  $R^3(x,y,z)$  will be transformed to joint space  $(\theta_1^*, \theta_2^*, \theta_3^*, \theta_4^*, \theta_5^*)$ , through the inverse kinematics computation, then is sent to the servo controller of the robotic manipulator. The tracking result of linear trajectory tracking is displayed in Fig. 13. The overall running time is 1.2 second and the tracking errors are less than 4mm. Similarly, the circular trajectory command generated by 300 equidistant segmented points with center (220,200,300) mm and radius 50 mm is tested again and its tracking result is shown in Fig. 14. The overall running time is 3 second and the trajectory tracking errors at each axis are less than 2.5mm. Figures 13~14 show the good motion tracking results under a prescribed planning trajectory. Therefore, experimental results from Figs. 11~14, demonstrate that the proposed FPGA-based motion controller for a robot manipulator ensures effectiveness and correctness.

### FPGA-based motion controller for robot manipulator

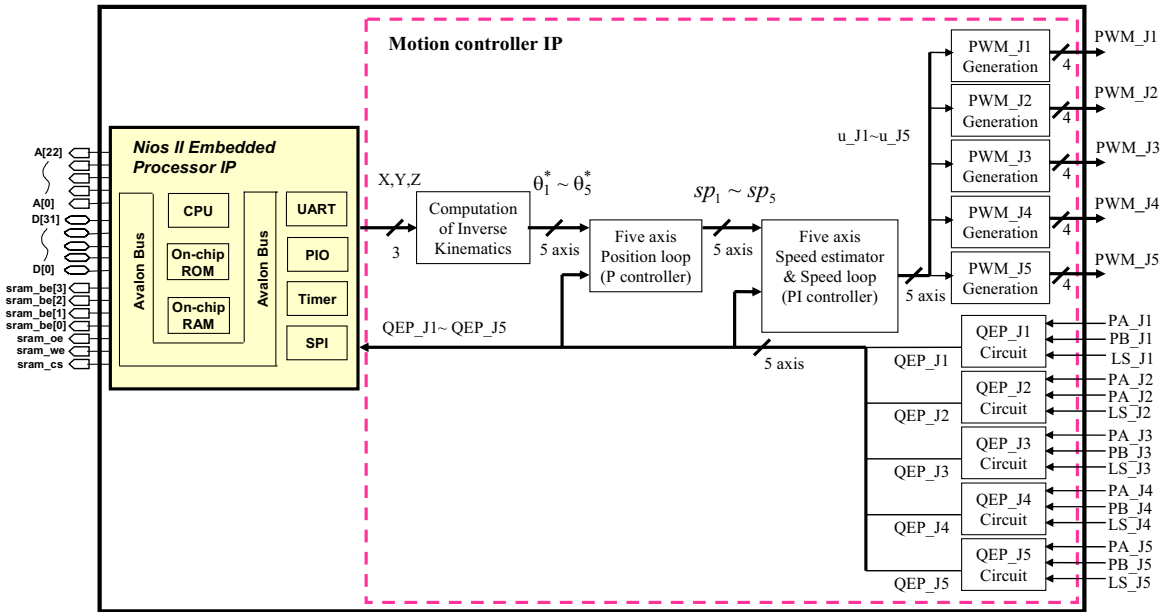


Fig. 3 Internal architecture of the proposed FPGA-based motion controller for robot manipulator

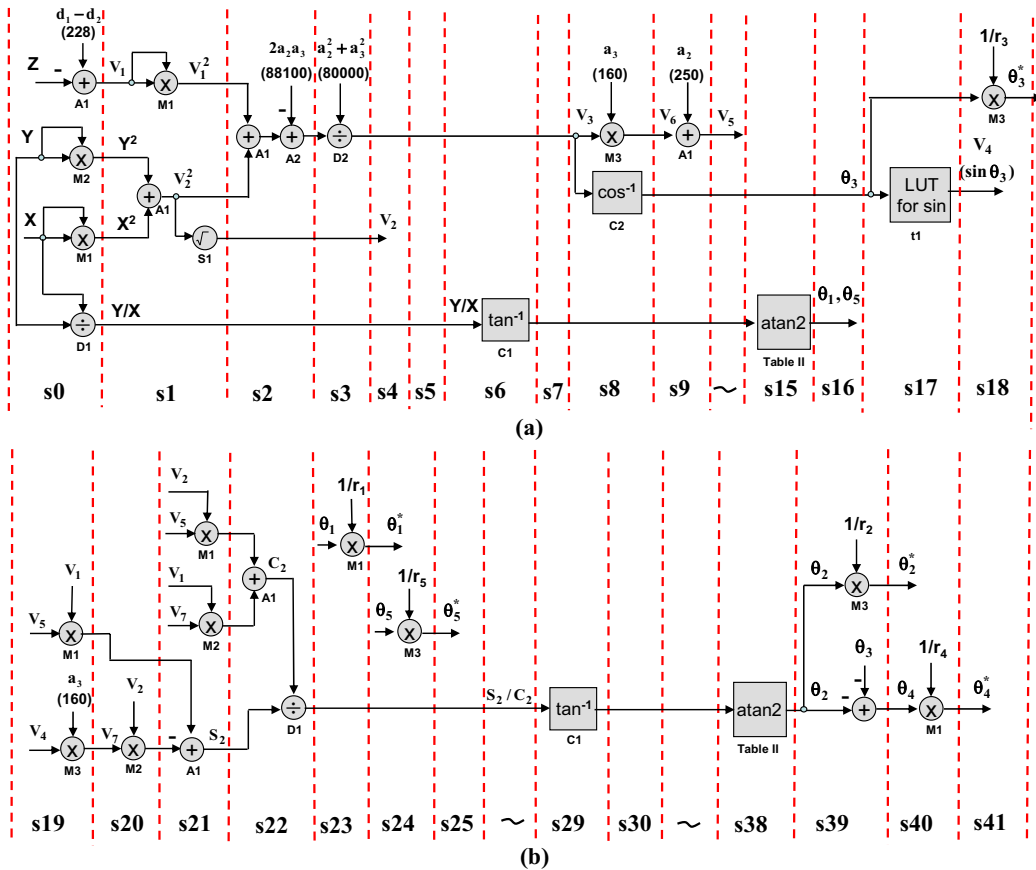


Fig. 4 Circuit of computing the inverse kinematics using finite state machine

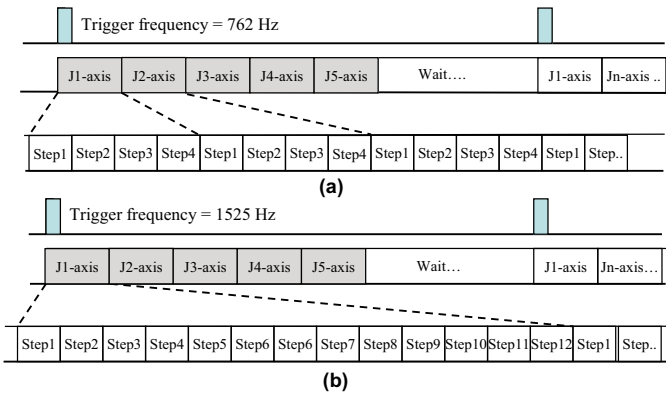


Fig. 5 Step control signal generated sequence at (a) position loop (b) velocity loop

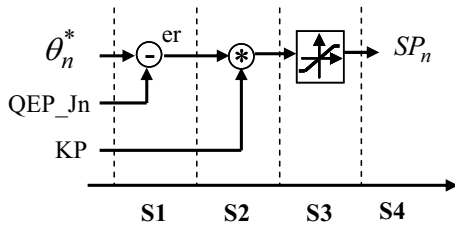


Fig. 6 P controller circuit in position loop of each axis

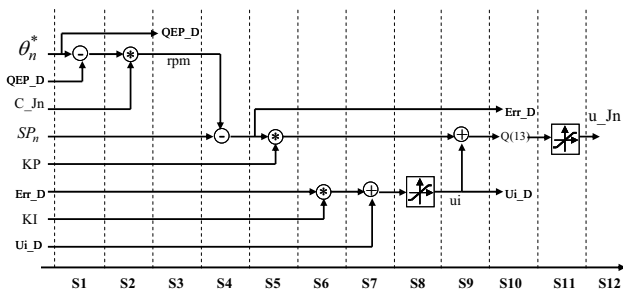


Fig. 7 Speed estimation and PI controller circuit in speed loop of each axis

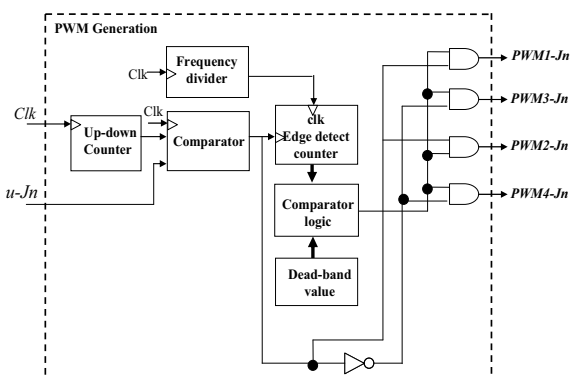


Fig. 8 PWM circuit

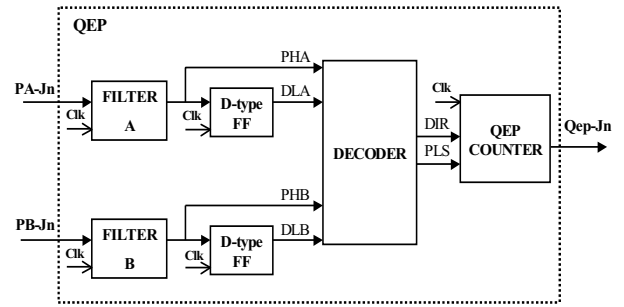


Fig. 9 QEP circuit

TABLE III  
Evaluation of utilities in FPGA

Component	Resource usage in FPGA (*ALMs)
Nios II Processor	2,481
Five sets of PWM circuits	316
Five sets of QEP circuits	285
Five axis position controller	206
Five axis speed controller	747
Circuit of inverse kinematics	6,994
Others	71
<b>Total</b>	<b>11,591</b>

\* 1 ALMs  $\cong$  2.5 Logic Elements

#### IV. CONCLUSIONS

The implementation of inverse kinematics and servo controller for robot manipulator using FPGA has been successfully demonstrated in this paper. The computation time to implement the inverse kinematics in hardware is only 840ns, but same formulations implemented by Nios II processor in software need 5.6ms. Furthermore, in this paper also presents a fully digital motion controller for robot manipulator, that the inverse kinematic computation, position and velocity controller, PWM generation and QEP circuits are all integrated and realized in a single FPGA chip. Finally, some experiments are also successfully validated, and the experimental results show a good performance.

#### ACKNOWLEDGMENT

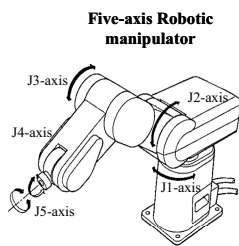
This work was supported by National Science Council (NSC) and Industrial Technology Research Institute (ITRI) of the R.O.C.

#### REFERENCES

- [1] M. Kabuka, P. Glaskowsky and J. Miranda, "Microcontroller-based Architecture for Control of a Six Joints Robot Arm," *IEEE Trans. on Industrial Electronics*, vol. 35, no. 2, 1988, pp.217-221.
- [2] G. Yasuda, "Microcontroller Implementation for Distributed Motion Control of Mobile Robots," in *Proceedings of International workshop on Advanced Motion Control*, 2000, pp. 114-119.
- [3] T.S. Li, S.J. Chang and Y.X. Chen, "Implementation of Human-like Driving Skills by Autonomous Fuzzy Behavior Control on an FPGA-

based Car-like Mobile Robot," *IEEE Trans. on Industrial Electronics*, vol. 50, no. 5, 2003, pp.867-880.

- [4] S.N. Oh, K.I. Kim and S. Lim, "Motion Control of Biped Robots using a Single-Chip Drive," in *Proceedings of the IEEE International Conference on Robotics & Automation*, 2003, pp. 2461~2469.
- [5] Y.S. Kung and G.S. Shu, "Development of a FPGA-based Motion Control IC for Robot Arm," in *Proceedings of the IEEE International Conference on Industrial Technology*, pp.1397-1402, Dec. 14~17, 2005.
- [6] M.A. Hannan Bin Azhar and K.R. Dimond, "Design of an FPGA Based adaptive Neural Controller for Intelligent Robot Navigation," in *Proceedings of the Euromicro Symposium on Digital System Design*, 2002.
- [7] X. Shao and D. Sun, "A FPGA-based Motion Control IC Design," in *Proceedings of the IEEE International Conference on Industrial Technology*, 2005, pp.131-136
- [8] R.Wei, X.H. Gao, M.H. Jin, Y.W. Liu, H.Liu, N. Seitz, R. Gruber and G. Hirzinger, "FPGA based Hardware Architecture for HIT/DLR Hand," in *Proceedings of the IEEE/RSJ International Conference on intelligent Robots and System*, 2005, pp. 523~528.
- [9] Y.S. Kung, P.G. Huang and C.W. Chen, "Development of a SOPC for PMSM Drives," in *Proceedings of the 47<sup>th</sup> IEEE International Midwest Symposium on Circuits and Systems*, 2004, vol. II, pp. II-329~II-332.
- [10] www.altera.com.
- [11] Altera, *Nios II Development Kit, Stratix Edition – Getting Started user Guide*, 2005.
- [12] L.W. Tsai, *Robot Analysis-The Mechanics of Serial and Parallel Manipulators*, John Wiley & Sons, Inc, 1999.
- [13] R. Schilling, *Fundamentals of Robotics – Analysis and control*, Prentice-Hall International Editions, 1998.



Name of Link	No.	Max. Working range (degree)	Encoder (ppr)	Gear ratio
waist	J1	300° (67416 pulse)	800	1:100
should	J2	130° (49311 pulse)	800	1:170
elbow	J3	110° (27958 pulse)	800	1:110
wrist pitch	J4	±90° (±17676 pulse)	380	1:180
wrist roll	J5	±180° (±20667 pulse)	380	1:110

Fig. 10 Robot manipulator and the related system parameters

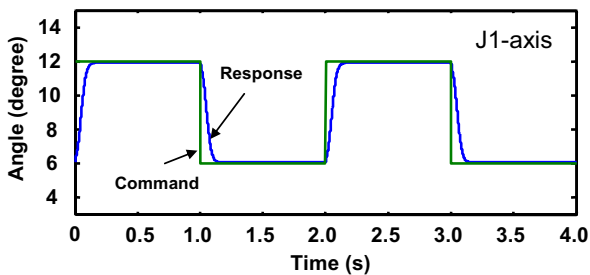


Fig. 11 Position step response at J1-axis of robot

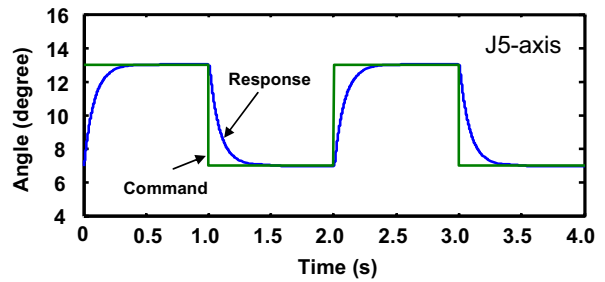


Fig. 12 Position step response at J5-axis of robot

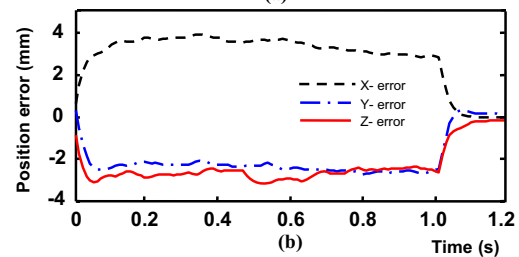
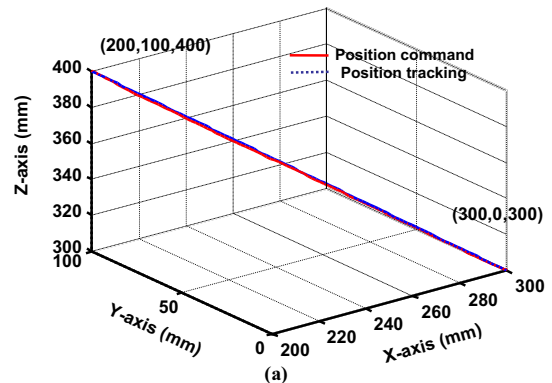


Fig. 13 Linear trajectory tracking

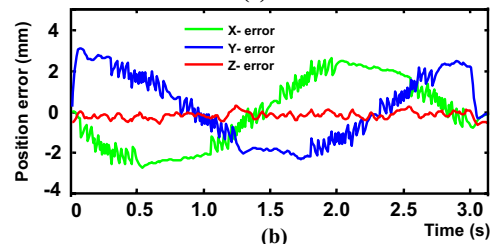
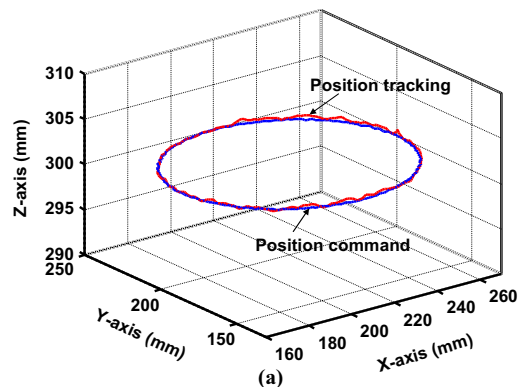


Fig. 14 Circular trajectory tracking