Third Prize

SOPC-Based Servo Control System for the XYZ Table

Institution: **Southern Taiwan University of Technology/Motor Engineering Research Institute**

Participants: **Dai Fuyu, Cai Xing'an, and Chen Jiasheng**

Instructor: **Ying-Shieh Kung**

Design Introduction

Electric power is mainly derived from a combination of an engine and a motor. The motor converts electrical power into mechanical energy, which is widely used in home appliances and industrial mechanical tools. When addressing specific applications, motor-driven tools generally need to make speed changes or have accurate positioning. In these applications, it is necessary to have a highly efficient servo-motor control to position the tools' movements accurately.

There are direct current (DC) and alternating current (AC) motors. DC motors used to be popular in the industry because of their simple controls—you only needed to control the armature voltage to vary the motor's speed. Because the motor's carbon brush and convector were mechanical components, they would produce sparks and cause damage when the motor was running, which was one major shortcoming of the DC motor. In addition, the DC motor posed a threat to the environment, had a short lifecycle, and was expensive to maintain.

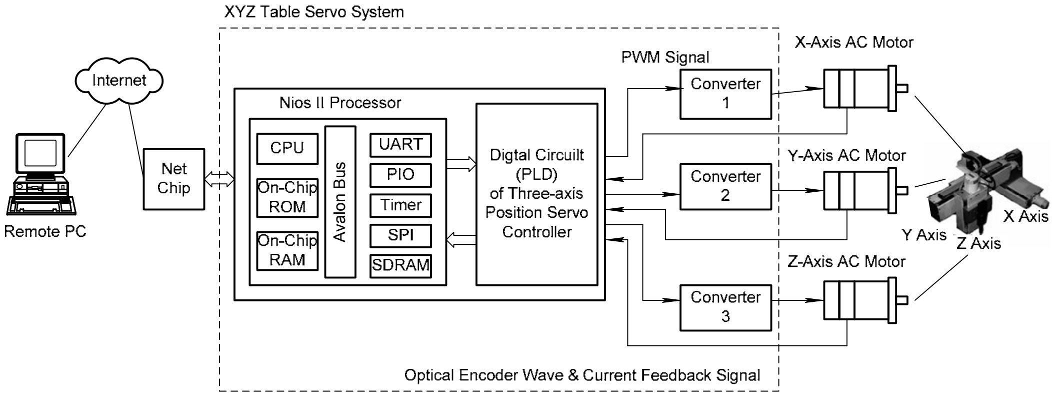
AC motors can be classified into three different types: the synchronous motor, the induction motor, and the reluctance motor. The stator and rotor of an AC motor are the only contact bearing components. The spinning of the rotor is caused by the stator's magnetic field, and needs more complicated control technology (such as the magnetic field guide control) to implement different movements. With the development of semiconductor control devices, the computation required for AC motor control is easily met. Because of this advantage, AC motors are very popular today.

This project was created to study and design an integrated chip of multi-axis AC permanent magnet synchronous servo motor control system using system-on-a-programmable chip (SOPC) concepts using an Altera® FPGA and the Nios® II embedded processor. We used the device to implement a three-axis XYZ table servo-control system, as shown in Figure 1.

The SOPC-based servo control system for XYZ table design mainly includes two modules: software and hardware. The software module is implemented using the Nios II processor, with programs handling communication between the control chip and PC, process control of three-axis XYZ table servo movement, and computation of movement tracking. Altera's FPGA implements the hardware module, which includes the functions to control the position of three motors on the three-axis XYZ table, a six-group PI controller algorithm computation, a three-group optical encoder signal-detection circuit, a three-group current-estimation circuit, a three-group vector-control coordinate conversion circuit, and a three-group space vector pulse width modulator (SVPWM) signal output.

Hardware digital circuits implement all position controllers of the three motors. We implemented the process control module in software because the computation is complicated and needs to be flexible. Further, the sampling frequency is not high. For instance, the moving track control is about 100 Hz. The position servo control of the three AC motors on the three-axis XYZ table must be implemented in hardware because the algorithm requires a faster execution speed (SVPWM frequency is 12 kHz, counter frequency is 3 - 4 MHz). Both the software and the hardware modules can run concurrently. In this way, you can improve the control feature of a three-axis XYZ table servo system. The three-axis XYZ table servo-control chip completed in this design offers digital control, improved system performance, and stability. The device also helps reduce the controller's size and cost. Figure 1 shows a block diagram.

Figure 1. Block Diagram of XYZ Table Servo Control System



Our design can be applied in a variety of applications including CNC computer lathe processes, electrical-discharge machines, engraving machines, professional-drafting machines, mold and metal surface treatment, and high-tech semiconductor surface technology treatment.

We previously implemented the integrated design of AC servo system by using Texas Instruments' 243DSK development board and Altera's FPGA, which combined a microprocessor and FPGA architecture. As this design used a double-chip architecture, it needed an extra network control chip and linkage socket in case of additional network control. Unfortunately, this design approach resulted in increased system module size and costs. We combated these issues by using the Nios II development board. It is compact, economical, has a more stable control system design, and features a modular design. In addition, the powerful hardware circuits of Nios II development board offer functions such as a network control chip and extended memory. Therefore, we designed SOPC hardware with the Nios II development board and managed to complete the design much faster than the previous effort.

The SOPC Builder environment makes it easy to use the Nios II hardware development kit to create the required Nios II processor functionality, develop the complicated logic circuits, and easily modify the computing parameters. In addition, by using design partition techniques, we judiciously implemented a few fixed mode and high-speed logic operations in hardware. Then, using the Quartus® II integrated development environment (IDE), we quickly implemented and verified our hardware circuit design. Taking full advantage of the high performance Nios II embedded processor and the abundant logic resources of Stratix® FPGA, we were able to quickly and easily implement our design.

Function Description

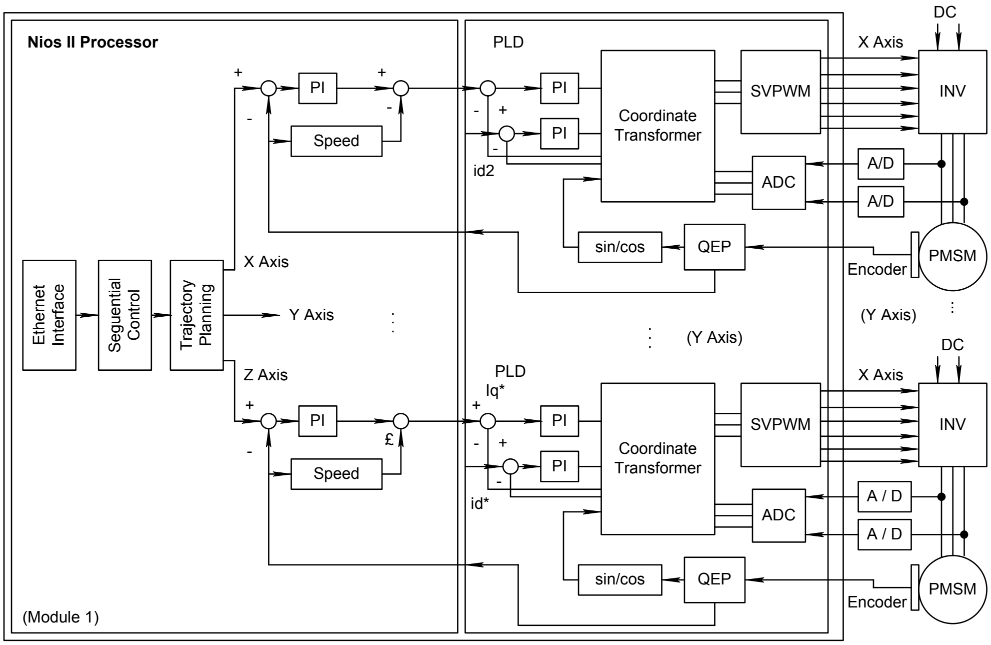
This section provides a functional description of the system.

Integrated Function of Three-Axis XYZ Table System Chip

Figure 2 shows the inner structure of three-axis XYZ table system chip. There are two modules in this FPGA-based system design. The software module is implemented using software programs in the Nios II processor and includes functions that handle the communication between the control chip and PC, process control of three-axis XYZ motor movement, and movement-track computation. The hardware module is implemented in the FPGA, with functions comprising the position control of three-AC motors on three-axis XYZ table, including six-group PI controller algorithm computation, three-group QEP detection circuit, three-group current estimation circuit, and three-group SVPWM signal output.

The position controllers of the three motors are all implemented using hardware digital circuits. In this way, the controller chip can receive the PC's remote control commands and send out the position control angle of three-axis motors after its computation by the software module. After being processed by the hardware module's three-axis XYZ position loop control circuit calculation, this data is then sent to the PWM signal of each axis for a precise control of three-axis motor shift to target position. The FPGA design approach can minimize the controller size, and the Nios II processor makes the system design flexible. In this way, we can improve the three-axis XYZ table servo control performance and reduce costs.

Figure 2. Inner Structure of Three-Axis XYZ Table Servo System Integrated Chip



System Architecture Description

The system, Figure 1, consists of:

- *Three-axis XYZ servo table*—This mechanism has three moving axis, and each axis is driven by a permanent magnet AC synchronous servo motor for linear movement, together with a ball screw. It features a maximum range of 300 mm. The power rating of the AC synchronous motor is 200 W, features Hall Sensor measurement of the magnetic pole position, and rotor or magnetic pole position measurement by incremental optical encoder (2,500 PPR); the transient current limit is about 10A (max). The rating speed is 2,500 rpm. The range of the ball screw is 5 mm/pitch.

- *Converter of three-group AC motor*—As shown in Figure 1, each group driver individually drives the AC servomotor on XYZ table. The power crystal of the driver is based on Toshiba's IGBT, and we also used Toshiba's TLP250 Photo integrated circuit (IC). The converter receives the PWM signal sent by the FPGA to drive the AC motor.

- *FPGA chip*—We used the Altera Stratix II EP2S60F672C5ES FPGA, featuring 24,176 ALM, 492 I/O, 36 DSP blocks, and a total of 2,544192 bits of on chip memory. We have also used one Nios II embedded soft core featuring 32-bit CPU, 16-Mbyte flash memory, 1-Mbyte SRAM, and 16-Mbyte SDRAM. These hardware resources can easily be adopted in the design of a three-axis XYZ table servo movement control chip.

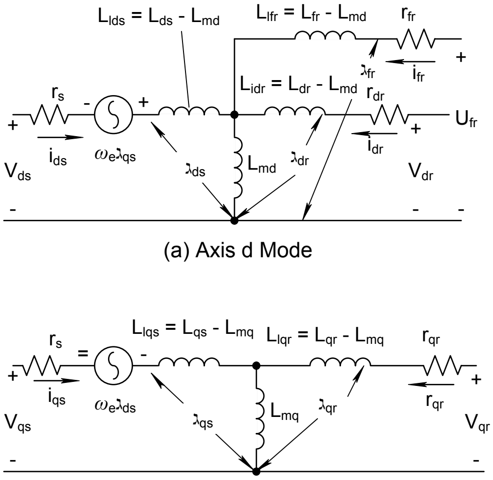
Detailed System Description

This section describes the system in detail.

Mathematical Model of a Permanent Magnet Synchronous Motor

The equivalent electric circuit of the permanent magnet synchronous motor is shown in Figure 3, where the voltage equation's reference coordinates are settled on the synchronous rotation coordinates.

Figure 3. Motor Axis d & q Mode



(a) Axis d Mode

(b) Axis q Mode

According to Figure 3, the axis d-q voltage can be shown as:

$$\begin{bmatrix} v_{qs} \\ v_{ds} \end{bmatrix} = \begin{bmatrix} r_s + sL_{qs} & \omega_e L_{ds} \\ -\omega_e L_{qs} & r_s + sL_{ds} \end{bmatrix} \begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix} + \begin{bmatrix} \omega_e \lambda_f \\ 0 \end{bmatrix}$$

The voltage is calculated to:

$$\frac{d}{dt}\begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix} = \begin{bmatrix} -\frac{r_s}{L_{qs}} & -\omega_e \frac{L_{ds}}{L_{qs}} \\ \omega_e \frac{L_{qs}}{L_{ds}} & -\frac{r_s}{L_{ds}} \end{bmatrix} \begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix} + \begin{bmatrix} \frac{v_{qs} - \omega_e \lambda_f}{L_{qs}} \\ \frac{v_{ds}}{L_{ds}} \end{bmatrix}$$

The motor torque produced by axis d flux and axis q current is:

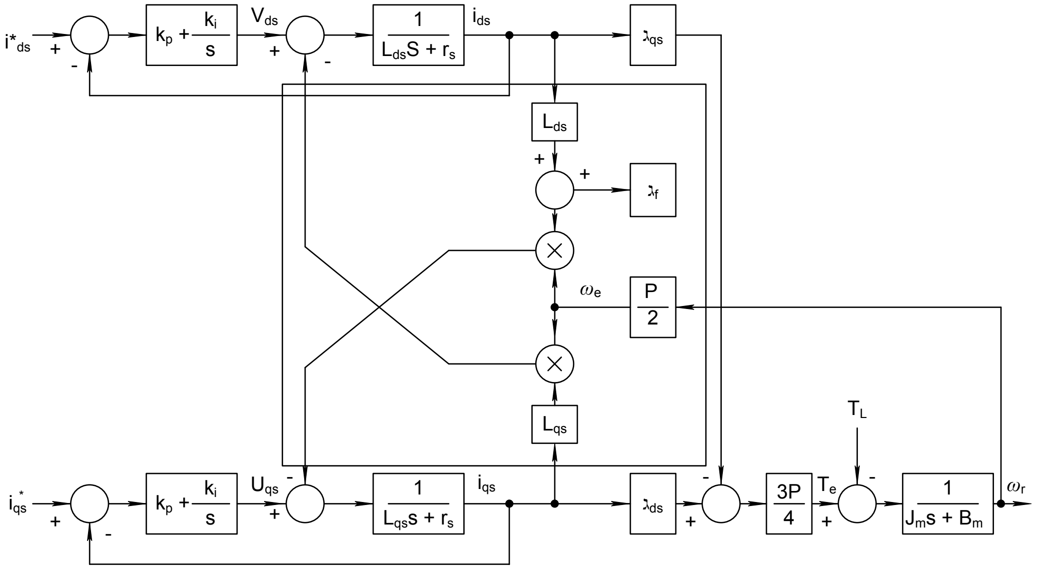
$$T_e = \frac{3}{2}\frac{P}{2}(\lambda_{ds} i_{qs} - \lambda_{qs} i_{ds})$$

Machinery dynamic equation obtained after loading is:

$$T_e - T_L = J_m \frac{d\omega_r}{dt} + B_m \omega_r$$

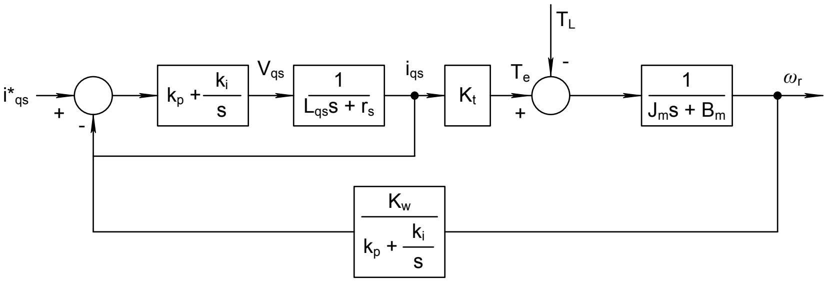
Combining the above three equations with the current controller of each axis, we can obtain the control block diagram during permanent magnet synchronous motor coupling, see Figure 4.

Figure 4. Control Block Diagram When Permanent Magnet Synchronous Motor Coupling



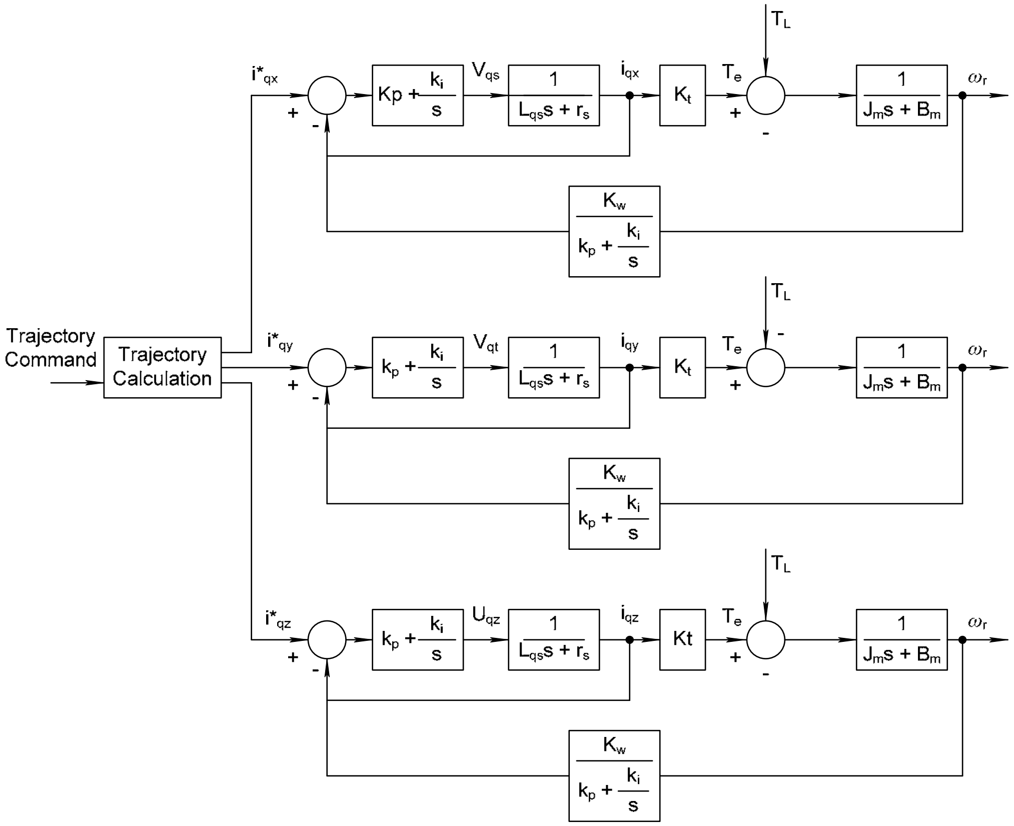
In this case, when the axis d current is set to 0, the motor torque or speed is only controlled by axis q current. This allows us to simplify the control block diagram as in Figure 5.

Figure 5. Diagram of Control Block



By the above permanent magnet synchronous motor control principle, the motor servo controller design is based on both the closed-circuit control and signal feedback by encoder of permanent magnet synchronous motor, making position control more accurate. Based on the above single-axis position control, we extended the system into a three-axis servo-moving controller (see Figure 6).

Figure 6. Three-Axis Servo Control Diagram



Point-to-Point Multi-Axis Track Planning & Design

For multi-position track planning, this design adopts a point-to-point track design. The point-to-point track design ignores the intermediate track and handles only the points of arrival and departure, to and from the destination. The design also takes into account the simultaneous departure and arrival, or featuring acceleration and deceleration functions upon departure and arrival. The input of point-to-point multi-axis design block diagram is equivalent to the rotation angle of each axis ($_1$, $_2$, $_3$), the maximum angular speed of each axis being (W_1, W_2, W $_3$), acceleration and deceleration times T_{acc} and sampling time $t_{d,}$ while output is the position command θ_r^* of each axis. The design takes the following computational steps:

1. Calculate total executing time

   T_1=MAX($_1$/W_1, $_2$/W_2, $_3$/W_3)

   Because total executing time should not be less than accelerating and decelerating time

   T_{acc}, therefore,

   T=MAX(T_1, T_{acc})

2. Revise total executing time as integral. Multiple of sampling time

   $N' = [T_{acc}/t_d]$ and $N = [T/t_d]$

   Among which, N is Interpolated point, and [ ] is Gauss function, therefore,

   $T' = N^* \, t_d$ and $T'_{acc} = N' * t_d$

3. Revise speed

   Order $\vec{L} \, \underline{\underline{\Delta}} \, (\Delta\theta_1, \Delta\theta_2, \Delta\theta_3)$

   $$\vec{W}' = \vec{L} / T'$$

4. Calculate accelerating and decelerating value

   $$\vec{A} \underline{\underline{\Delta}} (a_1, a_2, a_3) = \vec{W}' / T'_{acc}$$

5. Calculate intermediate position command

   (1) Accelerating segment:

   $$\vec{X}' = \vec{X}_0 + \frac{1}{2} * \vec{A} * t^2$$

   Among which, t =n* td and 0<n N1

   (2) Even speed segment:

   $$\vec{X}' = \vec{X}_1 + \vec{W} * t$$

   Among which, t =n* td and 0<n N2

   (3) Decelerating segment:

   $$\vec{X}' = \vec{X}_2 + (\vec{W}' * t - \frac{1}{2} * \vec{A} * t^2)$$

   Among which, t =n* td and 0<n N3

   And among $\vec{X}' \underline{\underline{\Delta}} (\theta^*_{1r}, \theta^*_{2r}, \theta^*_{3r})$, θ^*_{nr} is the position command value of the n-axis.

The design refers to Altera’s Stratix II EP2S60F672C5ES, which is used in constructing three-axis XYZ table servo movement control chip. The chip includes three-axis XYZ position loop control circuit, SVPWM circuit, QEP estimation circuit, and current estimation circuit. In this way, the controller can receive optical encoder signals of three-axis motor simultaneously, compare them with command position, and then send out PWM signal for each axis after calculation by Nios II and FPGA to drive the power crystal for precise control of three-axis motor shift to target position. In general, the application of the FPGA can not only minimize the controller size, but also improve the three-axis XYZ table servo control performance and reduce costs.

System Functions Accomplished by the Design

We used Visual Basic to develop three-axis XYZ table HCI movement and supervision software, and issue three-dimensional coordinates command through a remote PC. We were also able to set up the remote servo control mechanism through network transmission, and after track calculation by the Nios II processor we can obtain one point to another data in space, and rotation angle for each axis. The command for motor rotation position is obtained after track planning by position, computing by PI controller's position servo control, and the output of PWM command to calculate the PWM hardware circuit to drive motor driver. Then, after QEP hardware circuit computation, we were able to control motor to the required position. At this point, we sent back the sensor measurement for each axis position to Nios II CPU for conversion into three-dimensional coordinates, and transferred to HCI of remote PC through network, so the user can observe whether or not the motor has reached the required position.

Implementation of Hardware & Software Modules

The hardware and software modules are described as follows.

Hardware

Build a three-group power board, three-group motor driving circuit board, three-group analog-to-digital converter ADC converter circuit which includes the interface between three-axis motor driving board and FPGA chip, and power conversion circuit.

Software

First, we needed to write an HCI of PC that can transmit and receive through the network, and design a CPU core based on Altera's Stratix II EP2S60F672C5ES FPGA. This combination featured the necessary network transmission function and the required hardware interface for servo control, to implement a complete servo motor control chip. We developed the three-group SVPWM to produce the signal for driving AC motor circuit board, three-group QEP processor circuit to measure motor position. There are two functions involved in calculating the servo control position; the first function sets up network transmission/receiving mechanism, while the second, based on the required movement control equation for the practical use of three-axis XYZ table, writes the required response to movement. After resolving solution space vector, the space coordinates are sent back to the PC through the network to observe whether or not it has reached the required position.

Performance Parameters

For this experiment, we set the control interrupt frequency at 1 kHz in the Nios II processor, which is the most basic and important working frequency in the controller. The following is the measured interrupt time by Nios II CPU. In Figure 7, we can see that after several computing intensive steps, the Nios II CPU can precisely obtain the 1-kHz interrupt frequency required for the experiment, without any mismatch of control frequency. (1 stands for entering interrupt cycle, 0 stands for leaving interrupt cycle.)

Figure 7. Obtaining the Frequency

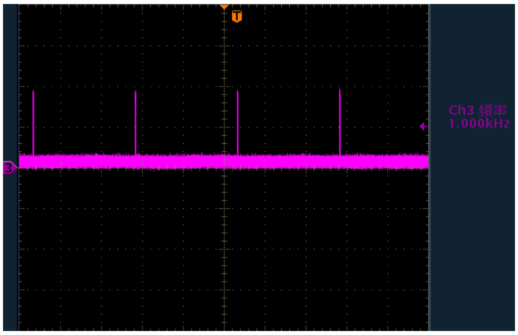


Figure 8 is a point-to-point echelon speed-change track after Nios II CPU computation. The blue line stands for the command and the green line stands for the feedback signal. The working frequency is 100 Hz after the interrupt frequency is issued, subject to acceleration, uniform velocity, and deceleration.

Figure 8. Echelon Track Designing Position Command & Feedback

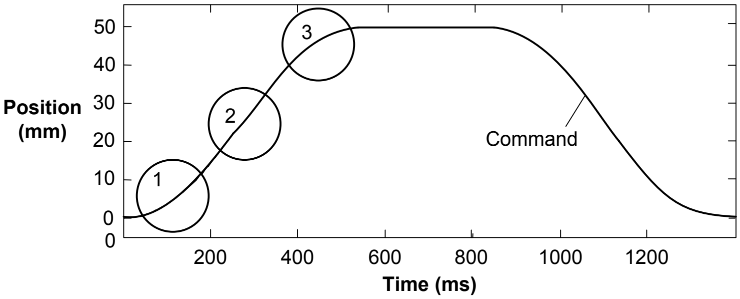
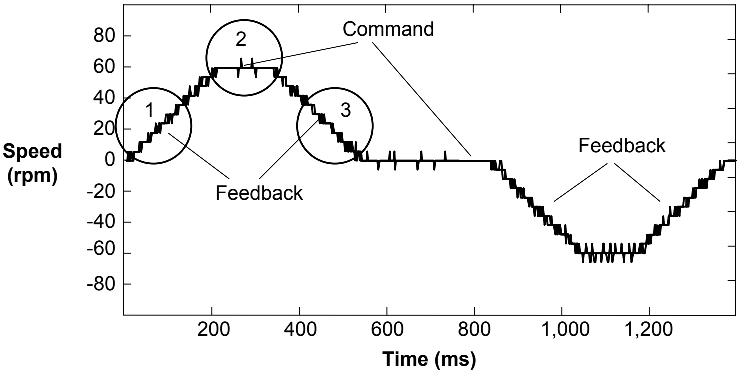


Figure 9 shows a designing position command with 5-cm radius circular track after calculations. The working frequency is 50 Hz after the interrupt frequency is removed.

Figure 9. Echelon Track Designing Speed & Feedback



Figures 10, 11, and 12 show the circular track position and its sine waveforms.

Figure 10. Circular Track Position Command

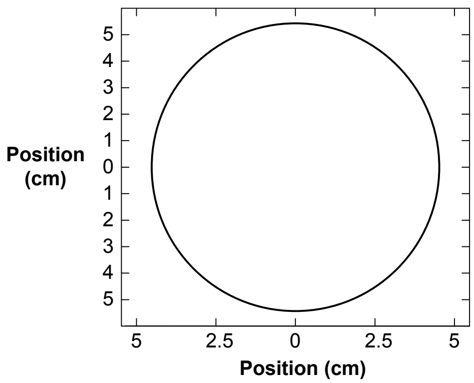


Figure 11. Sine Waveform (Command & Feedback) of Circular Track Position Command

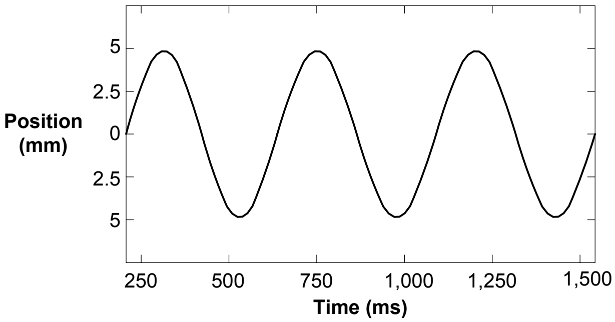
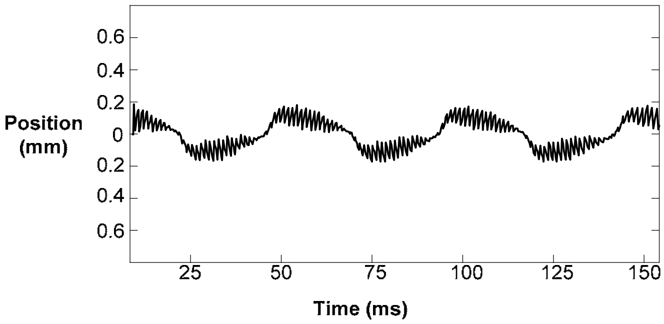


Figure 12. Sine Waveform (Position Error) of Circular Track Position Command



We can see from the above response identify that the response on the echelon track design is quite good, and the circular track position response can be seen from the sine waveform position error. The motor position control is also quite efficient.

Figure 13 shows the whole required track based on the computation of the CPU module's point-to-point track design, building the PI controller, speed, and position estimation module through the CPU. It draws Altera's logo using the XYZ table, with a working frequency set to 100 Hz, 50 Hz, and 25 Hz.

Figure 13. Required Track

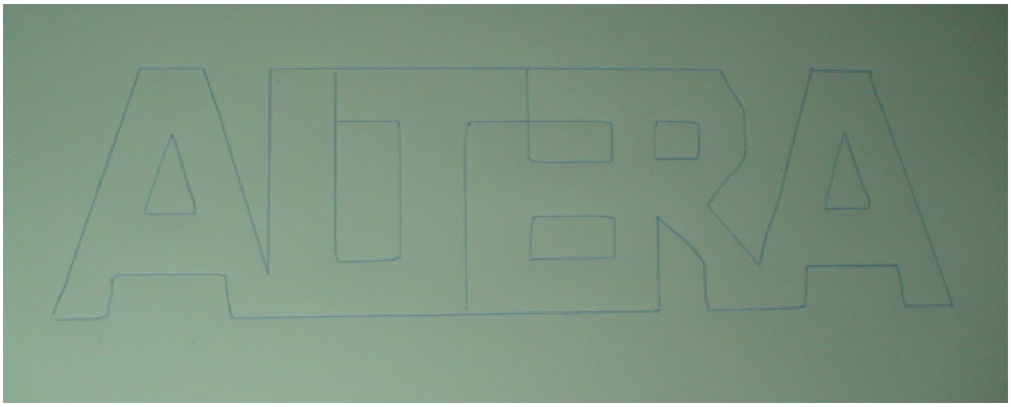
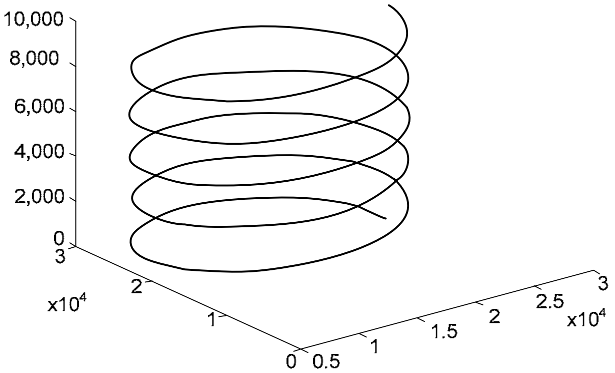


Figure 14 shows the 3-D circular track position command after computation by the Nios II CPU with the working frequency set to 50 Hz.

Figure 14. Circular Track Position Command



In this experiment, the Nios II processor was of crucial importance. The Nios II processor provided powerful debugging during the design process, which is an inseparable key function for system design. While we used the Nios II processor mainly for the controller design, we also used it to design multi-group PI controller, speed and position estimating module, 3-D circular track, point-to-point track, echelon speed-change track, and one-group 1-kHz interrupt program. Also, after filtering out the interrupt frequency, we obtained the various requires working frequencies.

Design Architecture

The three-axis XYZ table servo control system based is shown in Figure 15, and its experimental system is shown in Figure 16. The system consists of the following parts:

- *Three-axis XYZ table*—There are three moving axis, and each axis driven by permanent ac synchronous servo motor in linear movement together with a ball bearing guiding screw, as shown in Figure 15.

- *FPGA development board*—This board is the system core. We used Altera's Stratix II EP2S60F672C5ES FPGA to develop a control chip of the three-axis XYZ table.

- *Three-group ac motor converter*—The converter can receive the output PWM signal of control chip, and invert it into different voltages to control AC motor.

- *PC*—Develop supervision software for the man-machine interface.

Figure 15. Three-Axis XYZ Servo Table

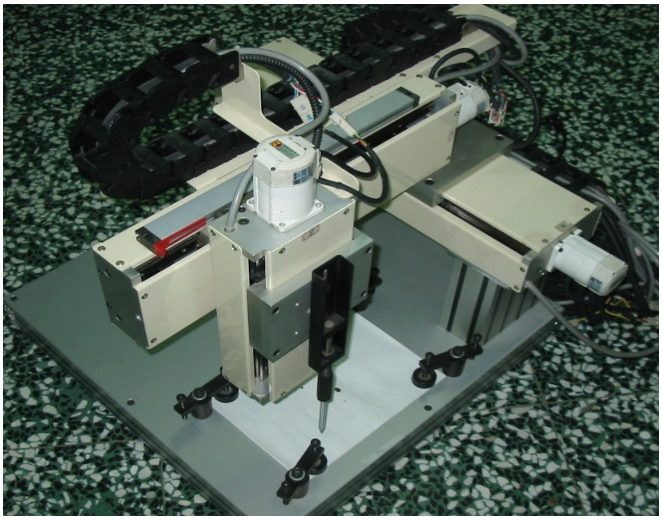
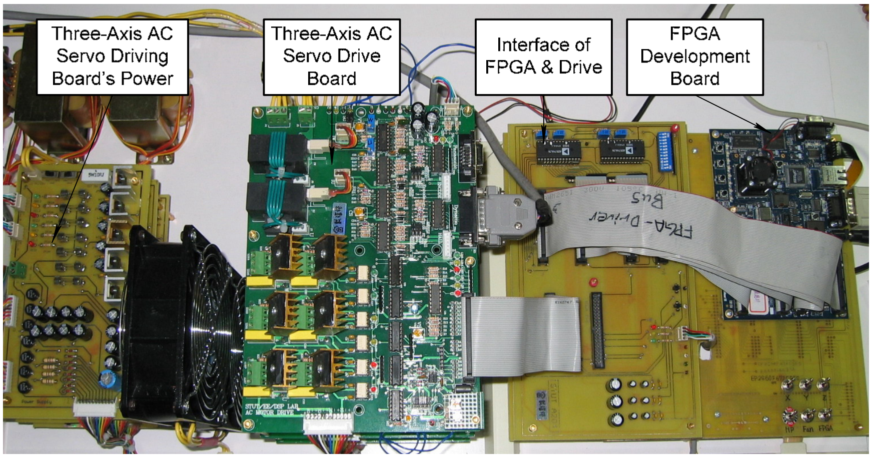


Figure 16. Three-Axis XYZ Table Servo Control Experimental System


Three-Axis AC Servo Driving Board's Power
Three-Axis AC Servo Drive Board
Interface of FPGA & Drive
FPGA Development Board

The inner hardware circuit of the three-axis XYZ table server system integrated chip is shown in Figure 17, which includes two modules. Module 1 is implemented in the Nios II processor by software, with functions including communication between the control chip and PC, process control of the three-axis XYZ table movement, and computation of the moving track. Module 2 is implemented in the FPGA, with the execution of three-axis servo controller for the table. The detailed circuit diagram of module 2 is shown in Figure 17, which includes six-group controller arithmetic, three-group QEP detecting circuit, three-group current estimating circuit, and three-group SVPWM signal output. Circuits are shown in Figures 17 and 18.

Figure 17. Inner Hardware Circuit of Three-Axis XYZ Table System Integrated Chip

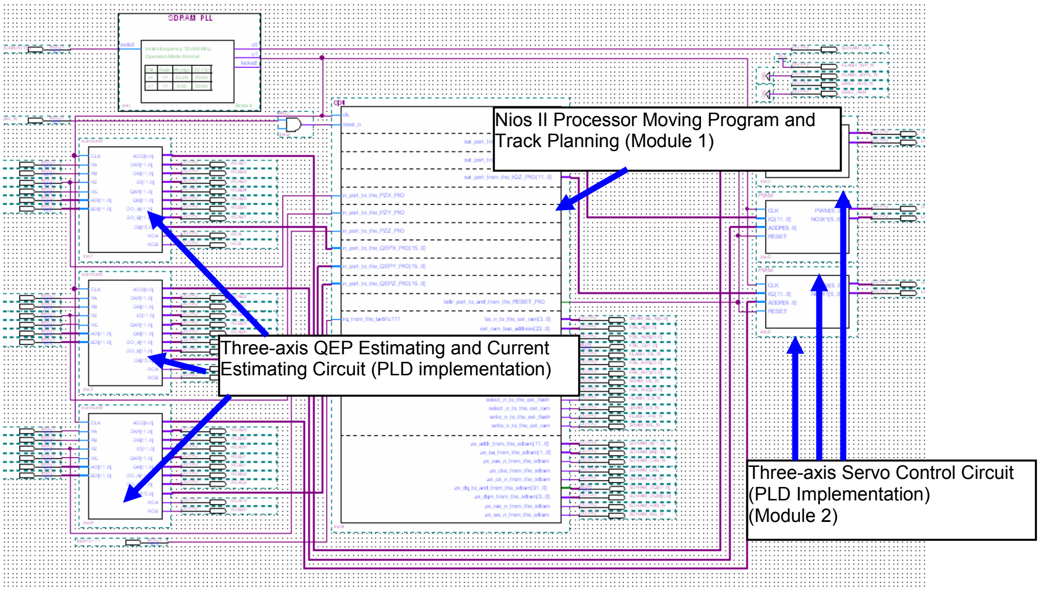

SDRAM PLL
Nios II Processor Moving Program and Track Planning (Module 1)
Three-axis QEP Estimating and Current Estimating Circuit (PLD implementation)
Three-axis Servo Control Circuit (PLD Implementation) (Module 2)

Figure 18. Three-Axis XYZ Table System Integrated Chip Inner Module 2—Three-Axis Servo Control Circuit

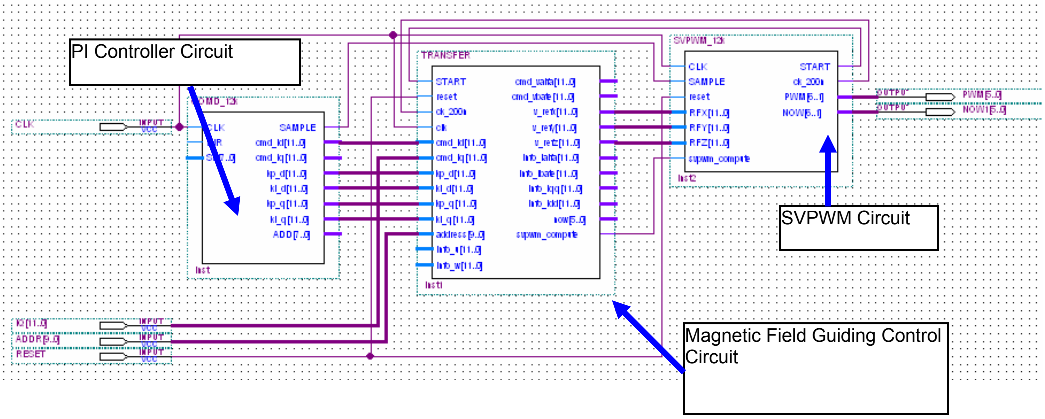


Figure 19 shows the proportion integral (PI) controller.

Figure 19. PI Controller Circuit Block Diagram

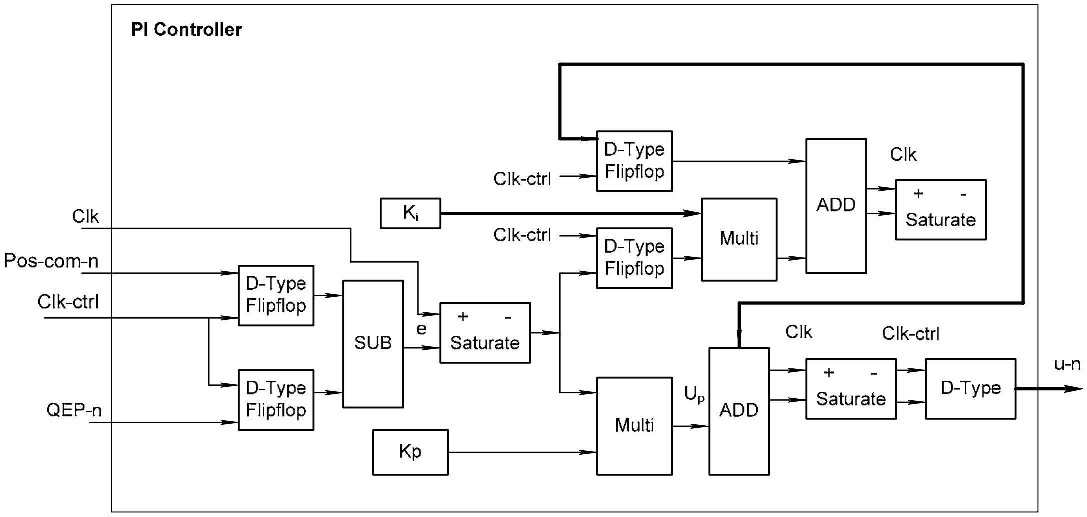


Figure 20 shows the QEP treatment circuit.

Figure 20. QEP Estimating Circuit Block Diagram

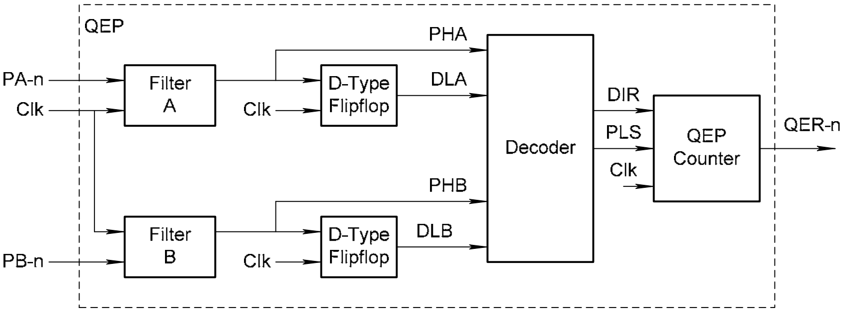


Figure 21 shows the SVPWM circuit.

Figure 21. SVPWM Circuit Block Diagram

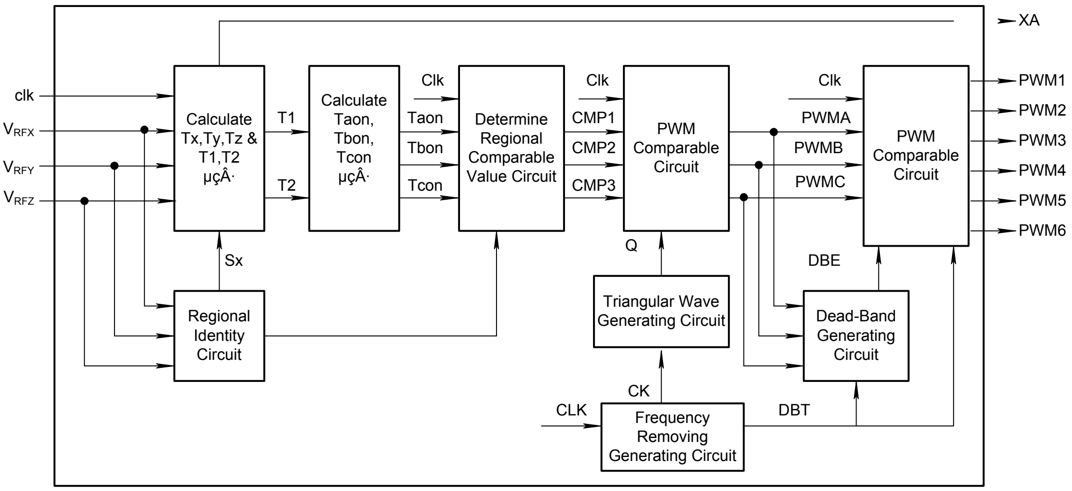
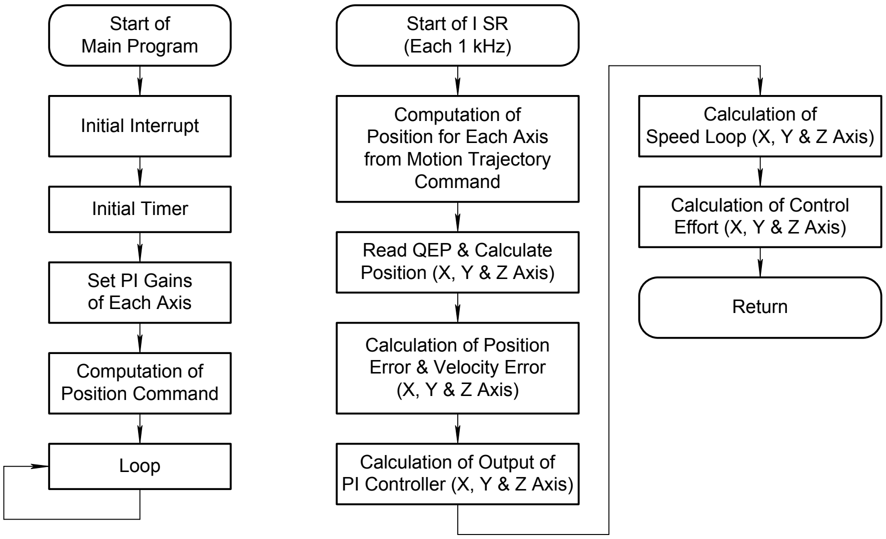

clk
V_{RFX}
V_{RFY}
V_{RFZ}
Calculate Tx,Ty,Tz & T1,T2 µçÂ·
T1
T2
Calculate Taon, Tbon, Tcon µçÂ·
Clk
Taon
Tbon
Tcon
Determine Regional Comparable Value Circuit
Clk
CMP1
CMP2
CMP3
PWM Comparable Circuit
Clk
PWMA
PWMB
PWMC
PWM Comparable Circuit
XA
PWM1
PWM2
PWM3
PWM4
PWM5
PWM6
Sx
Regional Identity Circuit
Q
Triangular Wave Generating Circuit
CK
CLK
Frequency Removing Generating Circuit
DBT
DBE
Dead-Band Generating Circuit

Figure 22 shows the software design flow chart for the control chip.

Figure 22. Nios II Three-Axis Servo Application Control Flow Chart


Start of Main Program
Initial Interrupt
Initial Timer
Set PI Gains of Each Axis
Computation of Position Command
Loop
Start of I SR (Each 1 kHz)
Computation of Position for Each Axis from Motion Trajectory Command
Read QEP & Calculate Position (X, Y & Z Axis)
Calculation of Position Error & Velocity Error (X, Y & Z Axis)
Calculation of Output of PI Controller (X, Y & Z Axis)
Calculation of Speed Loop (X, Y & Z Axis)
Calculation of Control Effort (X, Y & Z Axis)
Return

Design Methodology

The design methodology involved the following steps:

1. Develop the IP core of motor drive and design three-group AC servo motor drive using the VHDL language in the Quartus II integrated development environment (IDE), including magnetic field guiding control module, QEP estimation circuit, linear-optical scale position estimation, and current estimation circuit, ADC chip driver circuit, and SVPWM circuit.

2. Modify the Nios II processor, using SOPC Builder, to be the controller of the motor drive, design EMS memory unit flash and SDRAM, Avalon® tri-state bridge, system ID peripherals, JTAG UART, timer, and Avalon PIO. Next, design the reset address and exception address in flash memory and SDRAM, respectively. Finally, design the PLL for the CPU and SDRAM clock. Program the value of the motor drive obtained by the PIO using the Nios II processor, the output value of which is used for drawing waveforms in the MATLAB software for validation.

3. Design three-axis servo controlling system hardware which includes three-group power, three-group motor drive board, three-group motor required DC power, FPGA development board, and interface circuits for hardware including ADC circuit, linear optical scale signal receiving circuit, limit switch signal receiving circuit, and so on.

4. Design the motor controller and design one-group, 1-kHz interrupt program using the Nios II IDE. Develop many sub-programs: echelon track design, rotundity track design, PI controller, speed estimator, position estimator, and so on. First, calculate the required track by the main program, and then execute the PI controller, speed, and position estimator, moving the track program and the control program by a 1-kHz interrupt.

5. Integrate the three-axis servo control system, expand the original CPU to the Nios II processor to handle a three-axis servo control system, and combine the newly modified CPU with the three-axis motor drive and position estimating circuit. Next, output the previously written moving track program to the Nios II processor, whose motor drive circuit calculates the obtained command value and acquires the PWM signal. The signal is sent to the motor drive board via the FPGA development board and the necessary circuits to drive the AC Servo Motor.

6. Verify the program and check system whether the system is correct and efficient. Apply the previously designed program and control system on XYZ table to the Nios II processor-based system. Obtain the motor position and the speed feedback signal, output the command value compared with the feedback value, and verify and modify the controller's arithmetic parameter, until the controller's efficiency improves.

Design Features

We implemented our three-axis permanent AC synchronous servo motor control system design by using a complex arithmetic three-dimensional movement control. We converted the coordinates using the Nios II CPU and realized the remote control and monitoring through network mechanisms. While developing this project, we used the Nios II processor's powerful functions to quickly modify the system's computing parameters, verify its correctness, and were able to greatly reduce the design time.

Conclusion

This contest enabled us to develop a better understanding of the Nios II processor. By using the Nios II processor, we could easily design our system, which includes many embedded processors, on-chip and off-chip EMS memory, high-speed I/O ports, and network functions. The Altera development tools let us develop our own multi-functional IC quickly. Additionally, we could modify the CPU hardware at any time for multi-purpose development using the SOPC Builder tool. As for the Nios II IDE part, we hope to use the Nios II debug function to shorten the software development time significantly. Altera's ability to develop and update the Nios II IP and functions was extremely important. For example, using custom instructions we can accelerate the hardware computation speed, which can improve our system's efficiency. We hope that Nios II users across the globe will exchange ideas more frequently for the purpose of improving future designs. This exchange will promote the advantages of the Nios II processor and help to boost the development of SOPC technology.