Development of Chip-based Precision Motion Controller

Jung-Uk Cho* and Jae-Wook Jeon**

* School of Information and Communication Engineering, Sungkyunkwan University, Seoul, Korea
(Tel : +82-31-290-7237; E-mail: ichead@ece.skku.ac.kr)
** School of Information and Communication Engineering, Sungkyunkwan University, Seoul, Korea
(Tel : +82-31-290-7129; E-mail: jwjeon@yurim.skku.ac.kr)

Abstract: The Motion controllers provide the sophisticated performance and enhanced capabilities we can see in the movements of robotic systems. Several types of motion controllers are available, some based on the kind of overall control system in use. PLC (Programmable Logic Controller)-based motion controllers still predominate. The many peoples use MCU (Micro Controller Unit)-based board level motion controllers and will continue to in the near-term future.

These motion controllers control a variety motor system like robotic systems. Generally, They consist of large and complex circuits. PLC-based motion controller consists of high performance PLC, development tool, and application specific software. It can be cause to generate several problems that are large size and space, much cabling, and additional high coasts. MCU-based motion controller consists of memories like ROM and RAM, I/O interface ports, and decoder in order to operate MCU. Additionally, it needs DPRAM to communicate with host PC, counter to get position information of motor by using encoder signal, additional circuits to control servo, and application specific software to generate a various velocity profiles. It can be causes to generate several problems that are overall system complexity, large size and space, much cabling, large power consumption and additional high costs. Also, it needs much times to calculate velocity profile because of generating by software method and don't generate various velocity profiles like arbitrary velocity profile. Therefore, It is hard to generate expected various velocity profiles. And further, to embed real-time OS (Operating System) is considered for more reliable motion control.

In this paper, the structure of chip-based precision motion controller is proposed to solve above-mentioned problems of control systems. This proposed motion controller is designed with a FPGA (Field Programmable Gate Arrays) by using the VHDL (Very high speed integrated circuit Hardware Description Language) and Handel-C that is program language for deign hardware. This motion controller consists of Velocity Profile Generator (VPG) part to generate expected various velocity profiles, PCI Interface part to communicate with host PC, Feedback Counter part to get position information by using encoder signal, Clock Generator to generate expected various clock signal, Controller part to control position of motor with generated velocity profile and position information, and Data Converter part to convert and transmit compatible data to D/A converter.

Keywords: Motion Controller, Robot, Motor, Chip-based, FPGA, VHDL, Handel-C

1. INTRODUCTION

The motion control has emerged as one of the most dynamic technologies in manufacturing in the last decade. The electronic servo control system is promising to increase process speeds and throughputs by 50 percent or more, depending on the application. Additionally, motion controllers that combine with the high power processors and denser power devices are enabling more productive, versatile, reliable and predictable factory automation system. The Motion controllers provide the sophisticated performance and enhanced capabilities we can see in the movements of robotic systems. Several types of motion controllers are available, some based on the kind of overall control system in use. PLC (Programmable Logic Controller)-based motion controllers still predominate. The many peoples use MCU (Micro Controller Unit)-based board level motion controllers and will continue to in the near-term future.

These motion controllers control a variety motor system like robotic systems. Generally, they consist of large and complex circuits. PLC-based motion controller consists of high performance PLC, development tool, and application specific software. It can be cause to generate several problems that are large size and space, much cabling, and additional high coasts. MCU-based motion controller consists of memories like ROM and RAM, I/O interface ports, and decoder in order to operate MCU. Additionally, it needs DPRAM to communicate with host PC, counter to get position information of motor by using encoder signal, additional circuits to control servo, and application specific software to generate a various velocity profiles. It can be causes to generate several problems that are overall system complexity, large size and space, much cabling, large power consumption and additional high costs. Also, it needs much times to calculate velocity profile because of generating by software method and don't generate various velocity profiles like arbitrary velocity profile. Therefore, It is hard to generate expected various velocity profiles. And further, to embed real-time OS (Operating System) is considered for more reliable motion control [1-5].

In this paper, the structure of chip-based precision motion controller is proposed to solve above-mentioned problems of control systems. This proposed motion controller is designed with a FPGA (Field Programmable Gate Arrays) by using the VHDL (Very high speed integrated circuit Hardware Description Language) and Handel-C that is program language for deign hardware. This motion controller consists of Velocity Profile Generator (VPG) part to generate expected various velocity profiles, PCI Interface part to communicate with host PC, Feedback Counter part to get position information by using encoder signal, Clock Generator to generate expected various clock signal, Controller part to control position of motor with generated velocity profile and

position information, and Data Converter part to convert and transmit compatible data to D/A converter.

The contents of this paper are as follows. In section 2, explain structure and function of each block in proposed motion controller that is designed with a FPGA by using the VHDL and Handel-C. In section 3, experiment with real 1-axis robot system and show experimental result. The conclusion is described in the section 4.

2. CHIP-BASED MOTION CONTROLLER

The proposed chip-based motion controller has real benefits from ease of use, small size and space, and reduced system cost because all functions are integrated in a FPGA chip unlike former PLC-based and MCU-based motion controller. The structure of developed motion controller with FPGA is as in the Fig. 1.

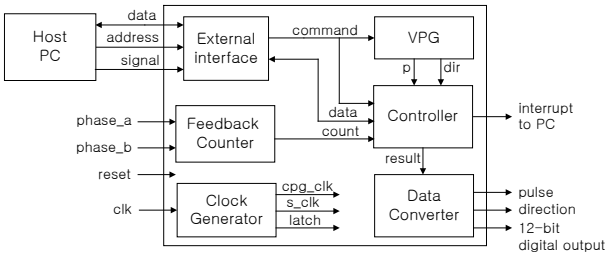


Fig. 1 The structure of Chip-based Motion Controller

The PCI Interface part is adopted to communicate with host PC. The host PC transmits command code to motion controller via 32-bit address and 16-bit data line in PCI bus. The Clock Generator makes various clock signals that are used to regulate timing of overall system. The Feedback Counter counts external encoder signal and transmits counted data to Control part in order to inform current position information of motor. The expected velocity profile is generated in velocity profile generator (VPG) part. The Controller part performs PID control with current commanded amount of movement *p* and rotation direction of motor *dir* form VPG part and current position information *feedback* form Feedback Counter. The result of PID control *result* is transmitted to Data Converter then it is converted into the pulse *pulse* and direction *direction* for step motor driver and 2-channel servo and 12-bit digital output *12-bit digital output* for D/A converter of analog servo and digital servo. The interrupt signal to host PC is needed to monitor real-time operation of motion controller. So, Host PC received data of command and position information of motor every interval of sampling time.

2.1 Velocity Profile Generator Part

The reduction of trajectory errors is important method that improves the performance of motor system. The expected various acceleration and deceleration velocity profiles enable to reduce trajectory errors. The velocity profiles are generated by using methods like polynomial selection and digital convolution in former times, but the polynomial selection method has the defect of long calculation time and the digital convolution method has the defect that only generates identical acceleration and deceleration velocity profiles [5], [6]. Therefore, in this paper, the proposed method enables to solve above-mentioned problems. Various acceleration and deceleration velocity profiles are generated in the short calculation time by using this method. The point of the method is that acceleration and deceleration rate factors is calculated and stored in advance [7], [8], [9].

The Fig. 2 shows internal structure of velocity profile generator (VPG) part. The former acceleration and deceleration methods don't generate various velocity profiles like arbitrary velocity profile. But this proposed method generates various velocity profiles that have each acceleration and deceleration properties including arbitrary configuration.

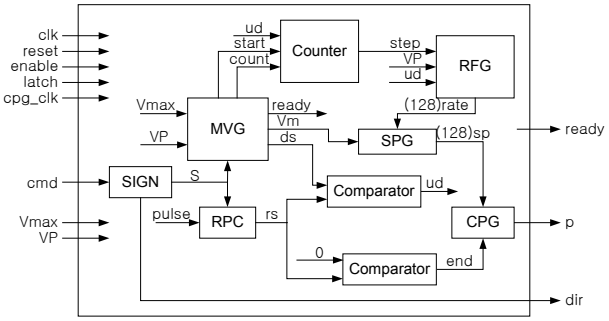


Fig. 2 The structure of Velocity Profile Generator part

The proposed acceleration and deceleration circuit of VPG part is designed by using this method. The proposed VPG part have inputs to which are signals indicative of the commanded amount of movement *cmd*, maximum velocity during the interval of sampling time *Vmax*, commanded type of velocity profiles *VP*, overall system clock *clk*, reset signal of overall system *reset*, enable signal about operation in VPG part *enable,* latch signal for data output *latch*, clock of CPG part *cpg_clk*.

The SIGN part converts commanded amount of movement into unsigned integer value *S* and rotation direction of motor *dir*. If commanded amount of movement is negative then the *dir* represents reverse rotation direction of motor. MVG (Maximum Velocity Generator) part calculates maximum velocity *Vm* and amount of deceleration movement *ds* in generated velocity profile. This part is design by using Handel-C because of integer division operation that is needed in processing. It is difficult to calculate integer division by using VHDL. Otherwise it is easy to calculate integer division by using Handel-C because it is software language for implementing algorithms in hardware straight form a C-based representation. The Counter part generates counted number *step* for RFG (Rate Factor Generator) part according *start*, *count*, and *ud.* The RFG part generates acceleration and deceleration rate factor of velocity profile during each interval of sampling time *rate*. The *rate* multiplied by 128 is *(128)rate* because the decimal fraction representation is impossible in hardware deign by using VHDL or Handel-C. The SPG (Sampling Pulse Generator) part generate *(128)sp* that represents the generated amount of movement during each interval of sampling time. The RPC (Remaining Pulse Calculator) part calculates *rs* that represents remaining amount of movement for starting deceleration. The *rs* is compared with *ds* in Comparator. It enables to know starting deceleration by *ud*. Also, the *rs* is compared with 0 in another Comparator. It enables to know stopping pulse output by *end*. The CPG (Commanded Pulse Generator) part generates *p* that represents commanded pulse during interval of sampling time. The *(128)sp* is divided by 128 by using shifting operation in CPG part. The CPG part has the well-known construction of a digital differential analyzer (DDA). The structure of CPG part is as Fig. 3. Specifically, the CPG part comprises a register in which the value *(128)sp* is set, an accumulator and an adder. The latter adds the contents of register and accumulator together each time a pulse *p* is generated, these pulses being generated at a constant frequency *cpg_clk,* and stores the

resulting sum in the accumulator. Owing the input from the adder, the accumulator produces overflow pulses as the output pulses *p* of the CPG part. If the accumulator has an n-bit construction, then letting the frequency of the pulses *cpg_clk* be given by:

$$cpg\_clk = Samplingclock \times 2^n \quad \text{(Pulse/sec)} \qquad (1)$$

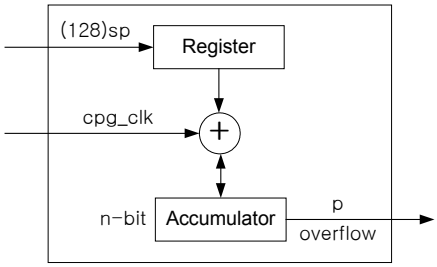


Fig. 3 The structure of Command Pulse Generator part

2.2 Controller Part

The Control part has structure as Fig. 4. This consists of PI (Pulse Integrator) integrating commanded pulse form VPG part during each interval of sampling time *ip* according to direction signal *dir* and pulse output *pulse*, Step Counter generating 4 clocks *step* during a sampling time for PID Controller part, PID Controller part controlling position of motor by PID control algorithm, IG (Interrupt Generator) part generating interrupt signal *interrupt to pc* to host PC every sampling time, and DT (Data Transmitter) part converting and transmitting compatible data to D/A converter and host PC according to latch signal *latch*. In the PID Controller part, the difference between commanded amount of pulses during interval of sampling time *ip* and current position information of motor *count* is used with P-gain, I-gain, and D-gain in PID control operation. The P-gain, I-gain, and D-gain values are multiplied by 256 because the decimal fraction representation is impossible in hardware deign by using VHDL or Handel-C. The equation of PID algorithm is as fellows.

$$V(n) = K_P E(n) + K_I \sum_{i=0}^{i=n} E(i) + K_D [E(n) - E(n-1)] \qquad (2)$$

The PID Controller part transmits result value of PID control operation *error*, integrated commanded amount of pulse *cmd*, integrated position information of motor *cnt* to DT part when calculation of PID control operation complete. DT part converts received data into compatible data for D/A converter and host PC and transmits converted data to Data Converter part and PCI Interface part. Then, IG part generates interrupt signal to PC *interrupt to pc* in order that host PC read data of integrated commanded amount of pulse *command* and integrated position information value of motor *feedback* every interval of sampling time via PCI Interface part. The result value of PID control operation *error* is multiplied by 256, so the *error* is divided by 256 by using shifting operation in DT part. The DT part transmits *result* into Data Converter part and *command* and *feedback* into PCI Interface part according to latch signal *latch* every interval of sampling time.

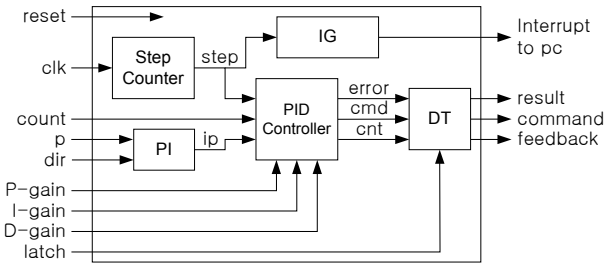


Fig. 4 The structure of Controller part

2.3 PCI Interface Part

The Fig. 5 shows internal structure of PCI Interface part.

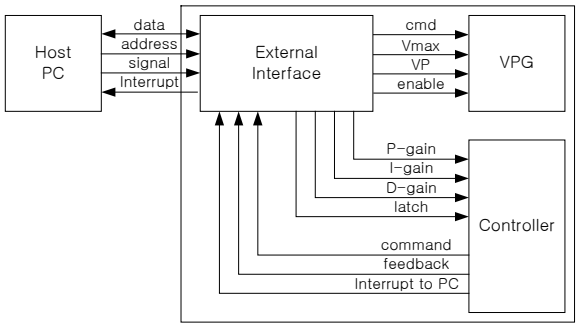


Fig. 5 The structure of PCI Interface part

The PCI Interface part is adopted to communicate with host PC. The host PC transmits command code to PCI Interface part via 32-bit address and 16-bit data line in PCI bus. PCI interface part decodes received data according to address and converts data into proper variables like *cmd*, *Vmax*, *VP*, *P-gain*, *I-gain*, and *D-gain*. Then the *cmd*, *Vmax*, and *VP* are transmitted into VPG part and the *P-gain*, *I-gain*, and *D-gain* are transmitted into Control part. The PCI Interface receives *feedback* and *command* from Controller part then transmits them into host PC. The interrupt signal *Interrupt* is transmitted host PC every interval of sampling time. Then host PC read data of *command* and *feedback* via address and data line in PCI bus. It enables to monitor real-time operation of motion controller. The developed PCI Interface part is developed based on PCI specification 2.1 that is announced PCI SIG (Special Interest Group) [10].

2.4 Clock Generator Part

The Clock Generator generates various clock signals like clock of sampling time *s_clk*, clock for CPG part *cpg_clk*, and control signal like latch signal to synchronize input and output timing of data in Controller part , VPG part, and DAC parts *latch*.

2.5 Feedback Counter Part

The Feedback Counter part receives encoder output signals *phase_a* and *phase_b* to obtain current position information and rotation direction of motor. The Feedback Counter part transmits data of current position information into Controller part every interval of sampling time according to latch signal *latch*. The encoder output signals often have some glitch signals that cause fatal error of Feedback Counter. So, digital delay filter is used to eliminate glitch. The Fig. 6 shows the structure of digital delay filter.

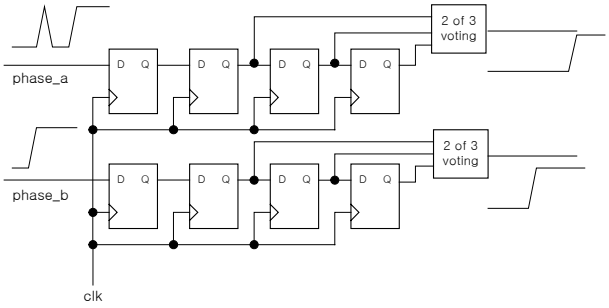


Fig. 6 The structure of digital delay filter

The digital delay filter bases 2-of-3 voting method by using four D-flip-flops. The 2-of-3 voting method is as fellows. The output signal is value of same two signals among former three input signals that have respective one clock delay by D-flip-flop. The glitch is eliminated in this process. The clock of D-flip-flop is changed according to maximum frequency of encoder. The proposed motion controller has Feedback Counter part that is 32-bit up-down counter with digital delay filter.

2.6 Data Converter Part

This proposed motion controller supports various output types. The Data Converter part receives result of PID control from Controller part and generates various output types to drive various motors. The pulse *pulse* and direction *direction* outputs are used for step motor driver and 2-channel servo and the 12-bit digital output *12-bit digital output* is used for D/A converter of analog servo and digital servo.

3. EXPERIMENT

The performance of proposed chip-based motion controller is checked in the experiment. The Fig 7 shows overall appearance of proposed chip-based motion controller system. This proposed controller is embodied in a FPGA device. This device is XC2S600E-6FG456C that is Spartan series FPGA in Xilinx, Inc. It has 600K system gates, 15,552 logic cells, and up to 329 I/O pins. The Table 1 is design summary that shows overall amount of used resource in overall device [11].

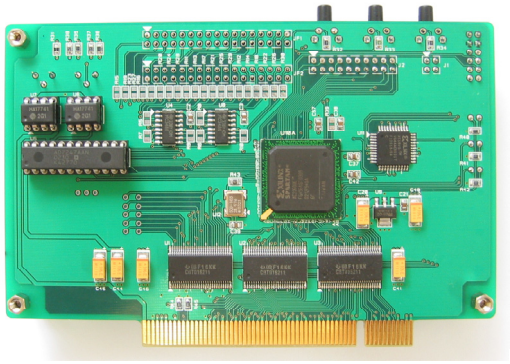


Fig. 7 The proposed chip-based motion controller

Table 1 The design summary

Design Summary

Number of errors: 0
Number of warnings: 14
Logic Utilization:
Total Number Slice Registers: 867 out of 13,824 6%
Number used as Flip Flops: 775
Number used as Latches: 92
Number of 4 input LUTs: 7,182 out of 13,824 51%
Logic Distribution:
Number of occupied Slices: 4,989 out of 6,912 72%
Number of Slices containing only related logic: 4,989 out of 4,989 100%
Number of Slices containing unrelated logic: 0 out of 4,989 0%
*See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs: 8,221 out of 13,824 59%
Number used as logic: 7,182
Number used as a route-thru: 1,038
Number used as Shift registers: 1
Number of bonded IOBs: 89 out of 325 27%
IOB Flip Flops: 35
Number of Tbufs: 64 out of 7,104 1%
Number of GCLKs: 2 out of 4 50%
Number of GCLKIOBs: 2 out of 4 50%

Total equivalent gate count for design: 79,954
Additional JTAG gate count for IOBs: 4,368
Peak Memory Usage: 145 MB

In order to perform motion control of 1-axis motor by using proposed motion controller, the configuration of experiment is as in the Fig. 8. It consists of motion controller, host PC, AC servo motor, and analog AC servo. The command is transmitted into proposed motion controller after setting internal variables in the host PC then commanded pulses are generated in proposed Velocity Profile Generator part. The calculated position error by using commanded pulses and feedback data is used in PID control operation in the Controller part. The result of PID control operation is converted and transmitted into analog AC servo then motor is drove into expected position by output of AC servo.

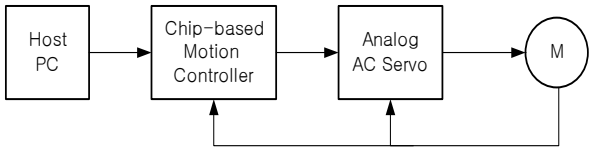


Fig. 8 The configuration for experiment

The proposed motion controller support GUI (Graphic User Interface) program for convenience of users. The users only click the button after inputting parameters in the window of GUI program in order to drive motor into expected position. The development tool of this GUI program is Visual C++ in Microsoft Corporation. The Fig. 9 shows executed configuration of GUI program. The Status part shows the current executed command. In the Jog part, the users can do simple driving of motor like acceleration and deceleration. The users click the Move button after inputting parameters and selecting type of velocity profile in the Move part, and then the proposed controller generates commanded acceleration and deceleration velocity profile and executes precision motion control according to parameters.

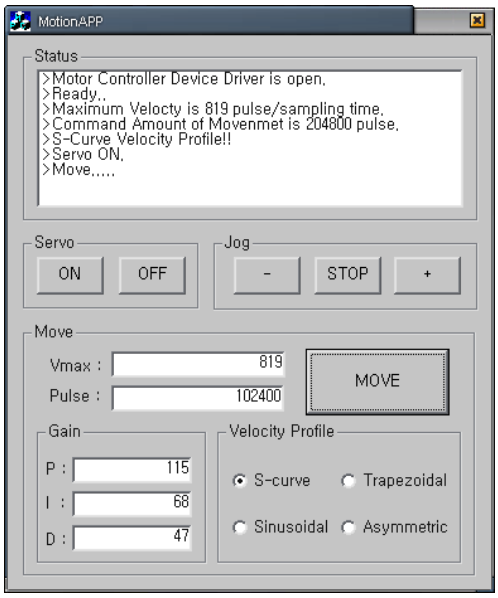


Fig. 9 The executed configuration of GUI program

The external D/A converter is needed to drive motor by using analog AC servo in the experiment. The Fig. 10 shows the configuration of D/A converter circuit for analog AC servo. The 12-bit digital output *12-bit data* is transmitted into D/A converter with control signals like *wr*, *cs_a*, *cs_b*.

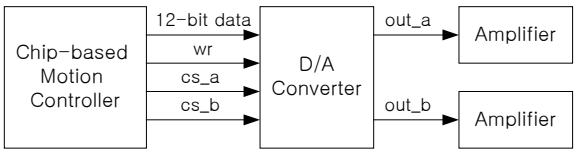


Fig. 10 The diagram of D/A converter output

In order to change digital signal into analog signal, this proposed motion controller adopts AD7247 in Analog Devices, Inc. It has 12-bit resolution and two-separated DAC. The output voltage ranges of DAC are selected among 0~+5V, 0~+10V, -5~+5V. The DAC output is amplified by OP-AMP because the range of servo input voltage is –9~+9V. The operation method of D/A converter is as Table 2.

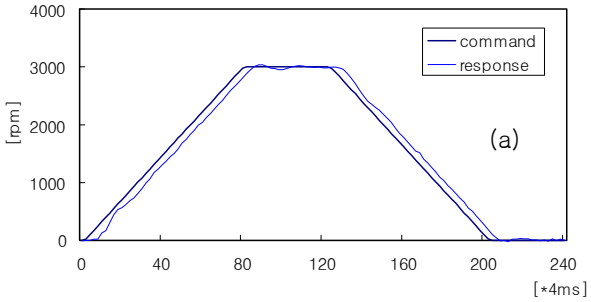
Table 2 The operation method of D/A converter

CSA	CSB	WR	Function
-	-	1	No Data Transfer
1	1	-	No Data Transfer
0	1	0	DACA Latch Transparent
1	0	0	DACB Latch Transparent
0	0	0	Both DAC Latches Transparent

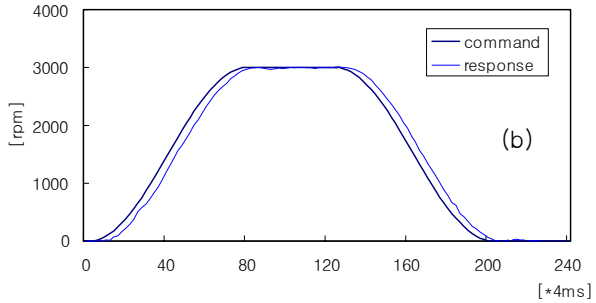
The velocity type AC servo driver CSA11-012SR12R and AC servo motor have properties like rated output 30Watt, rated rotational frequency 3,000rpm, rated torque 1.0 kg-cm, resolution of encoder 4096pulse/rev in this experiment [10]. The load of motor is round block that has properties like radius 3.9cm, thickness 2.5mm, weight 0.1108kg, moment of inertia 3.8kg-cm^2. The proposed motion controller generates expected various acceleration and deceleration velocity profiles like trapezoidal, s-curve, sinusoidal, and asymmetric. The proposed motion controller has variable sampling time by user define and each acceleration and deceleration time is variable. The sampling time is 4ms and the maximum acceleration time is 80ms in this experiment. The Table 3 is the equations of various acceleration and deceleration velocity profiles. The Fig. 11 shows result of experiment about various velocity profiles that are generated by using proposed chip-based motion controller.

Table 3. The equations of various velocity profiles

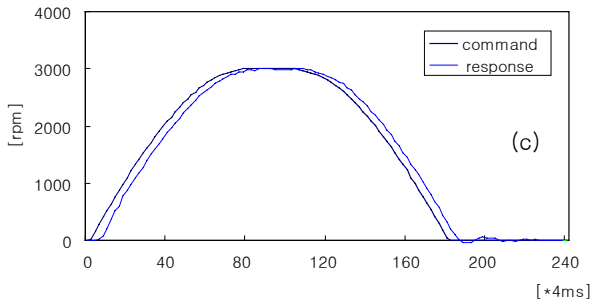
	Trapezoidal Velocity profile
Acceleration $(1 \le u < n_a)$	$f_a(u) = V_m \frac{u}{n_a}$
Deceleration $(n_d > u \ge 1)$	$f_d(u) = V_m \frac{u}{n_d}$
	S-curve Velocity profile
Acceleration $(1 \le u < n_a)$	$f_a(u) = \frac{V_m}{2}(\sin(\pi(\frac{u}{n_a} - \frac{1}{2})) + 1)$
Deceleration $(n_d > u \ge 1)$	$f_d(u) = \frac{V_m}{2}(\sin(\pi(\frac{u}{n_d} - \frac{1}{2})) + 1)$
	Sinusoidal Velocity profile
Acceleration $(1 \le u < n_a)$	$f_a(u) = V_m(\sin(\frac{\pi}{2} \times \frac{u}{n_a}))$
Deceleration $(n_d > u \ge 1)$	$f_d(u) = V_m(\sin(\frac{\pi}{2} \times \frac{u}{n_d}))$
	Symmetric Velocity profile
Acceleration $(1 \le u < n_a)$	$f_a(u) = V_m(\sin(\frac{\pi}{2} \times \frac{u}{n_a}))$
Deceleration $(n_d > u \ge 1)$	$f_d(u) = \frac{V_m}{2}(\sin(\pi(\frac{u}{n_d} - \frac{1}{2})) + 1)$



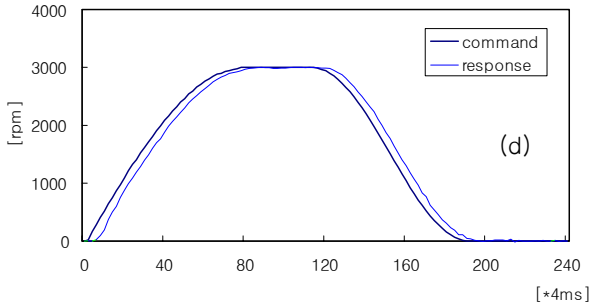
(a) The trapezoidal acceleration/deceleration velocity profile



(b) The s-curve acceleration/deceleration velocity profile



(c) The sinusoidal acceleration/deceleration velocity profile



(d) The asymmetric acceleration/deceleration velocity profile

Fig. 11 The results of experiment about velocity profiles.

4. CONCLUSION

The former motion controllers have many restrictions to generate various velocity profiles. The restrictions are overall system complexity, large size and space, much cabling, large power consumption and additional high costs. In this paper, the solution is proposed to replace former motion controller for industry robot and CNC machine and embodied in the chip-based motion controller with a FPGA. The proposed chip-based motion controller generates various acceleration and deceleration velocity profiles including arbitrary configuration. Also, it has real benefits from ease of use, small size and space, and reduced system cost because all functions are integrated in a FPGA chip unlike former motion controller.

It is expected that this developed chip-based precision motion controller can be easily applied to wider range of practical areas.

REFERENCES

[1] D. S. Khalsa, "High Performance Motion Control Trajectory Commands Based on The Convolution Integral and Digital Filtering," Proceedings of Intelligent Motion, pp. 54-61, Oct. 1990.

[2] R .Nozawa, et al. "Acceleration/Deceleration Circuit", United States Patent, Patent Number 4,554,497, Nov. 19, 1985.

[3] Koren Y., Computer Control of ManufacturingSystems, McGraw-Hill Inc. 1988.

[4] H. Inaba et al., “Method and Apparatus for Controlling the Acceleration and Deceleration of a Movable Element without Abrupt Changes in Motion”, United States Patent, Patent Number 4,555,758, Nov. 26, 1985.

[5] D. I. Kim, J. W. Jeon, and S. Kim, ”Software Acceleration/Deceleration Methods for Industrial Robots and CNC Machine Tools,” Mechatronics, Vol. 4, No. 1, pp. 37-35, 1994.

[6] J. W. Jeon, and Y. Y. Ha, “A Generalized Approach for the Acceleration and Deceleration of Industrial Robots and CNC Machine Tools,” IEEE Trans. On Industrial Electronics, Vol. 47, No.1, pp. 133-139, Feb. 2000.

[7] J. W. Jeon, and Y. G. Kim, ”FPGA Based Acceleration and Deceleration Circuit for Industrial Robots and CNC Machine Tools,” Mechatronics, Vol. 12, Issue 4, pp. 635-642, May. 2002.

[8] J. W. Jeon, ”Method for Controlling the Traveling Path of a Robot during Acceleration and Deceleration,” United States patent, Patent Number: 5,373,439, Dec. 13, 1994.

[9] J. W. Jeon, ”An Efficient Acceleration for Fast Motion of Industrial Robots,” Proceedings of 1995 IEEE 21st IECON, Orlando, FL, pp. 1336-1341, Nov. 6-10, 1995.

[10] PCI Local Bus Specification. Revision 2.1, PCI Special Interest Group, USA, 1995

[11] Spartan-IIE Complete Data Sheet (All four modules), Version: 2.1, Xilinx Inc., Jul. 9, 2003

[12] Servo Driver and Power Controller User’s Manual, Samsung Electronics Co., Ltd., 1993