國立交通大學

電機學院 電機與控制學程

碩士論文

應用順滑模態控制於微型音圈馬達摩擦力補償之研究

Friction Compensation of a Mini Voice Coil Motor by Sliding Mode Control

研 究 生：蔡慶隆

指導教授：林錫寬 博士

中華民國九十八年七月

應用順滑模態控制於微型音圈馬達摩擦力補償之研究
Friction Compensation of a Mini Voice Coil Motor by Sliding Mode Control

研 究 生：蔡慶隆　　Student：Ching-Lung Tsai

指導教授：林錫寬 博士　　Advisor：Dr. Shir-Kuan Lin

國 立 交 通 大 學

電 機 學 院　電 機 與 控 制 學 程

碩 士 論 文

A Thesis

Submitted to College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electrical and Control Engineering

July 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年七月

應用順滑模態控制於微型音圈馬達摩擦力補償之研究

學生：蔡慶隆　　指導教授：林錫寬 博士

國立交通大學　電機學院　電機與控制學程碩士班

摘　　要

本論文提出一順滑模態控制器，用來補償應用於照相手機 CCM 模組之內嵌導桿微型音圈馬達之非線性摩擦力，以達精密之位置控制。有效的補償線性摩擦力，此音圈馬達就可不需要使用潤滑油，如此便可提升其成本效能與可靠度效能。此音圈馬達之機構全行程僅 0.35㎜ 或甚至更短，其非線性摩擦力是一非匹配式的干擾，且其控制精度是微米之要求。使用本論文提出之控制器，再加上對順滑模態之狀態方程式做極點配置，即使在有非匹配式的干擾的情況下，不管穩態誤差之設計目標有多小，皆可達成。至於所提出之控制器之性能，首先搭配一音圈馬達之物理模型，用電腦模擬的方式來驗證，接著用實驗結果來證明所提出之控制器能夠避免傳統 PI 控制器所導致之 stick-slip 震盪。實驗結果亦證明馬達的到位速度與重現性優於一般的規格要求，並且驗證把順滑模態控制器的符號函數換成飽和函數可避免切跳現象。

Friction Compensation of a Mini Voice Coil Motor by Sliding Mode Control

Student : Ching-Lung Tsai Advisor : Dr. Shir-Kuan Lin

Degree Program of Electrical and Computer Engineering
National Chiao Tung University

Abstract

This thesis presents a sliding mode controller to compensate for the nonlinear friction of mini voice-coil motor (VCM), which is embedded with guide pins, for position control in compact camera module (CCM) application. With the effectiveness of compensating the nonlinear friction, the lubrication oil can be removed from the VCM, and the performance of cost and reliability then can be improved. In the VCM, the actuator's full stroke is only 0.35mm or less, nonlinear friction force falls into mismatched condition and um-level resolution is required. With the proposed controller, arbitrarily small steady state errors can be achieved by pole placement of the sliding mode state equation. The performance is firstly simulated with a physical VCM model. The experimental results are then presented to demonstrate the proposed controller is able to avoid stick-slip oscillation that usually yields when using a classic controller, such as PI controller. The experimental results also show that the performance of settling time and repeatability is better than nominal requirement and the undesired chattering can be remedied by replacing sign function with saturation function in the proposed control law.

Acknowledgement

The preparation of this important document would not have been possible without the help of so many people in so many ways, hard work and endless efforts of a large number of individuals and institutions. I would like to express my gratitude to all of them, realizing that those whom I owe the most I cannot thank enough, and that the things for which I am most grateful, I cannot put into words.

First of all, I would like to thank my advising professor Shir-Kuan Lin for his guidance and supervision, providing valuable review, comments, and suggestions during the finalization of the thesis.

Also, my gratitude is devoted to Dr. Mi-Ching Tsai, Dr. Ching-Chih Tsai and Dr. Jyh-Yeong Chang, the committee members during the oral examination. Thanks for having read a draft of this thesis and having made their precious opinion and helpful suggestions. I felt deeply honored that they can be my thesis committee members.

Besides, I would like to thank professor Ti-Chung Lee for his guidance, valuable advice and technique support. Professor Lee always tells me his knowledge thoroughly and generously.

On the other hand, I would like to thank my classmates, Shu-ting Chuo, Yi-Shien Chou, Chien-Chih Chang for their spiritual comments and encouragement.

Finally, I would like to sincerely acknowledge my parents. They always believed in me and encouraged me to do everything I wanted to do. I could not be here without them. I heartily thank my dear wife, whose love and continued support enabled me to overcome the frustrations which occurred in the process of writing this thesis. I am lucky to meet her in my life.

Ching-Lung Tsai

Table of Content

List of Tables

List of Figures

Chapter 1 Introduction

1.1. Research background and recent development

Compact camera module (CCM) has embedded in Mobile phone to provide camera function for years. The pixel size of CCM's image sensor has increased from 0.3M in the early stages to 5M presently. User's demand for better image quality of those cameras with image pixel size larger than 3M triggers CCM makers to build auto-focus (AF) function into high-end CCM.

AF function is established by an actuator, which carries lens, plus an image sharpness identifier (ISI). The actuator carries lens to different positions in the optical stroke and stop at the position with best image sharpness according to ISI's instruction. Camera's focus range can be from 10cm to infinity with the AF function. The opposite of AF is fixed-focus (FF). The focus range of FF CCM is specified for macro (10cm to 40cm), or infinity (90cm to ∞). It means the image quality is poor at macro if the FF CCM is specified for infinity. Under this condition, the high-pixel image sensor will not achieve the expected performance. That is why AF is strongly recommended for high-end CCM today.

Due to pursuing thin profile for mobile phone, CCM is required to be small. This limits the actuator solutions for the AF CCM. MEMS, piezo and mini voice-coil motor (VCM) are the available solutions in the market. The strengths of MEMS solutions are more precise positioning, better repeatability and lower power consumption. The weaknesses of MEMS solutions are poor reliability and higher cost. Piezo solutions also have the strength of low power consumption, but they have concerns about reliability and cost. VCM is the most popular solution for AF CCM due to its compact size, cost competitiveness and satisfactory performance. The size of VCM is down to $8.5 \times 8.5 \times 0.46 mm^3$ with 0.35mm full stroke for now and will become smaller in the future.

VCM solutions can be separated to two groups. One called tradition VCM, which builds in springs and adopts open-loop control. Its positioning is settled when the Lorenz force

yielded by the current of coils and the spring-force yielded by deformation are balanced. The other is closed-loop control VCM, which builds in position sensor and requires closed-loop controller. One weakness of the traditional VCM is that its spring would malfunction in continuous video mode. In this mode, the VCM may continuously work with max spring deformation. In the meantime, coils are heated up by the continuous driving current. And the heat is transferred to the spring because spring and coils are connected. The constant deformation plus the heat makes the spring not be able to restore to original position when the driving current is removed. On the contrary, closed-loop VCM will not have this problem because it does not embed with spring. Closed-loop VCM also provides better performance in transient response and positioning.

One kind of the closed-loop VCMs, as shown in Figure 1, supports good tilt angle performance by implementing guide pins ((b) of Figure 1), which directly contacts with actuator and results in nonlinear friction force ((g) of Figure 1). A well-known solution for decreasing the effect of nonlinear friction force is applying lubrication oil on the guide pins. But the optical system would fail if it is polluted by oil. So that taking care of the lubrication oil is an extra burden for manufacturing CCM. Furthermore, the effect of the nonlinear friction force will become more significant when the size of VCM is shrunk smaller (which is an inevitable trend). [13] even achieves power saving purpose by designing the static friction force to hold actuator at arbitrary position with zero supplied current. This characteristic is welcome by portable devices, such as CCM, because of power saving consideration. But it does add challenges on controller design because the effect of nonlinear friction is increased. So that developing a cost-effective friction compensation scheme with satisfactory performance is crucial for the future success of the mini VCMs embedded with guide pins.

Friction compensation has been studied for years in other fields of application, but it is new to study friction compensation scheme on the mini VCM applied to CCM. A comprehensive survey of friction compensation schemes is presented in [1]. In which, type A) and B) solutions are suitable for cost-sensitive applications because of its limited calculation burden. [11] proposed a nonlinear proportional controller with bang-bang force in specified region to compensate the sticking force, where [12] demonstrated a look-up table position controller, that has higher gain when the position error is smaller and lower gain when the position error is larger, to prevent stick slip oscillation. [9] developed adaptive model

following control (AMFC) to overcome the loading variation, and [10] considered an anti-windup PI controller, incorporated with the disturbance observer, to control the VCM.

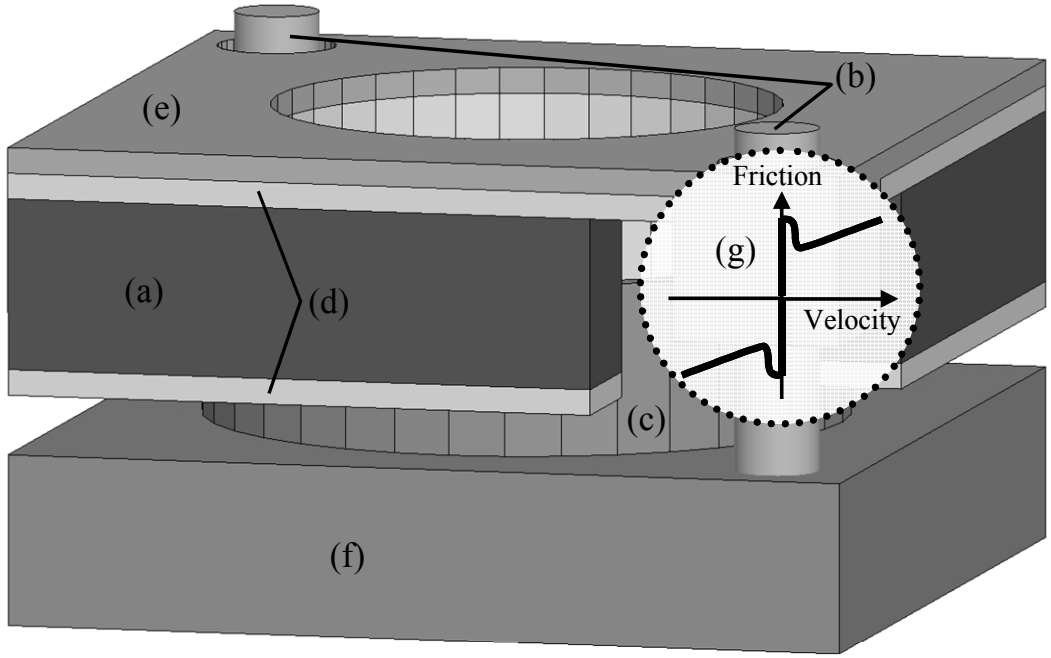


Figure 1 : Guide pins directly contacted with actuator result in significant nonlinear friction force.

1.2. Research motivation, objective and approach

The worldwide sales volume of mobile phones is over one billion in 2008, in which the high-end models all build in AF CCM. If closed-loop mini VCM solution can be more competitive in performance and cost, then its market share can be increased in AF CCM. Or even be able to convince middle-end (main stream) mobile phones to adopt AF solutions. Furthermore increase the market share of the mobile phone embedded with AF CCM. Then the whole vertical solution providers, such as VCM manufacturers, controller chip suppliers and AF CCM makers, will benefit from it.

We intend to improve the competitiveness of closed-loop mini VCM by developing a simple and effective control scheme to compensate the nonlinear friction between actuator and guide pins. Our goal is to remove the lubrication oil from the VCM. This can save the

cost of materials and manufacturing process. This also can improve the reliability of the VCM because lubrication oil is sensitive to temperature.

In this thesis, a sliding mode controller is developed to compensate the mismatched nonlinear friction force of the VCM, which does not apply lubrication oil on the guide pins. The controller only uses the information of static friction force of the VCM. With pole placement of the sliding mode state equation, steady state position error is predictable and is able to be controlled to be arbitrarily small. To the best of our knowledge, this kind of approach has not yet been studied and applied to the mini VCM system. The simulation results prove that the developed algorithm has better performance than classic controller, and the experiment results demonstrate the stick-slip oscillation is avoided. We also present an experimental environment (Figure 2), including PC motherboard, in which control algorithm is executed, and FPGA board, in which digital driver (full bridge) with synchronous current sampling [4] is implemented. The FPGA board connects to the motherboard through parallel-ATA (PATA) cable. Position feedback, current feedback and controller output traffic on the cable with standard PATA PIO protocol. This experimental platform provides the best cost/performance ratio comparing to others, such as ARM-base development board. Its high calculation power is especially suitable for advanced algorithm study. With this experimental environment, a more effective design flow for developing controller ASIC is also presented.

1.3. Thesis Organization

This thesis is organized as follows. In section II, physical VCM system modeling is described. In section III, a sliding mode control law for friction compensation is proposed, while simulation results are demonstrated in section IV. In section V, experimental results are provided with the brief implementation of PC-based experimental environment and design flow of controller ASIC. Finally, conclusions are presented in section VI.

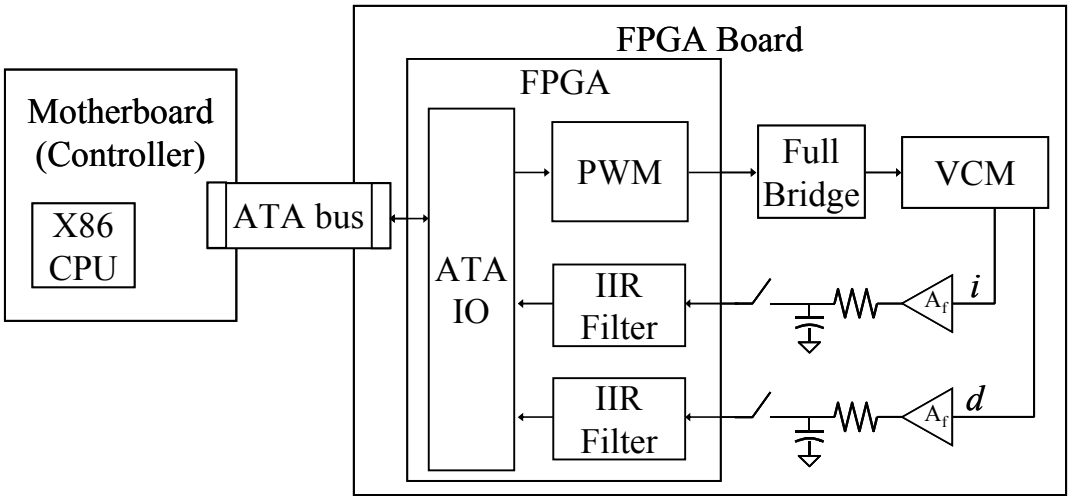


Figure 2 : Block diagram of experimental environment, including PC motherboard and FPGA board. FPGA board connects to PATA port of motherboard.

Chapter 2 Mathematical Model of the VCM

2.1. VCM introduction

VCM is a linear DC motor. It consists of two separate parts; the magnetic housing and the coil. Its architecture is very similar to that of loudspeaker, as shown in Figure 3. That is why its name is leaded by “voice coil”. The motor’s moving direction depends on the polarity of applied voltage on the terminals of the coil. Generally, constant force constant is designed in the full stroke range, such that the yielded force is proportional to the applied current on the coil. This characteristic makes VCM be easier for control by comparing to other motors. It does not need commutation. This non-commutated motor construction increases reliability. In most VCM applications, the load is directly coupled to the motor. This allows for fast acceleration. VCM itself offers unlimited resolution. The application resolution is constrained by the position sensor or the bit-length of the ADC used for position signal discretization.

VCM is very suitable for the applications which require high resolution and stroke is not long. Besides the emerging application on AF CCM, VCM has been widely adopted in many different kinds of applications, such as shown in Figure 4: the optical pick up head of DVD ROM and head stack of hard disk drive. Although VCM technology is old, but there are always new applications for it.

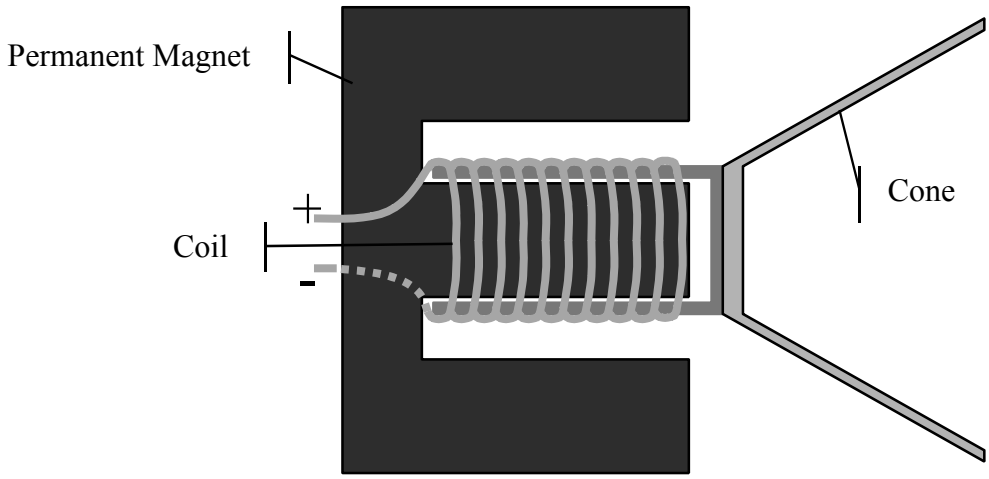
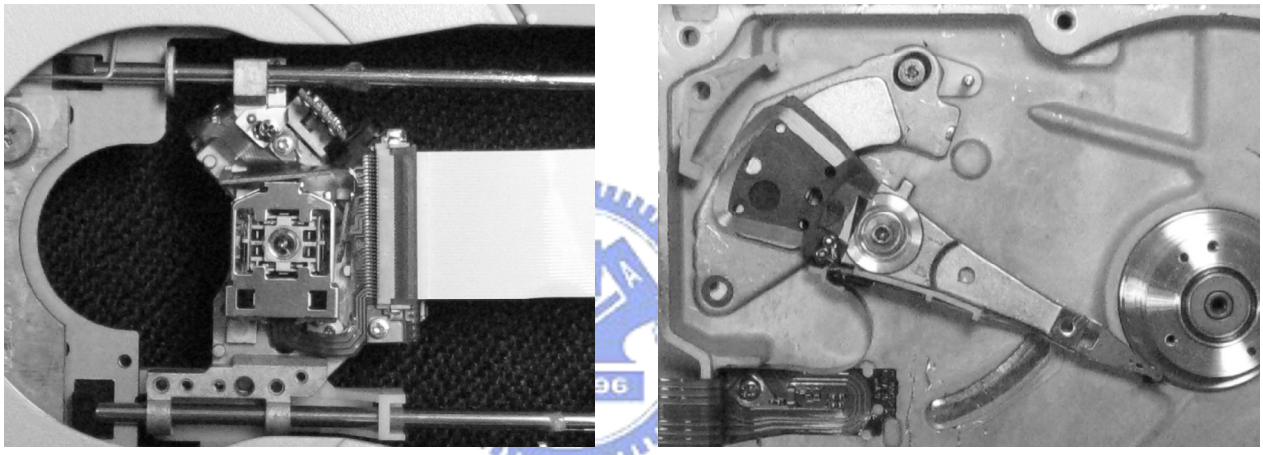


Figure 3 : Cross-section of loudspeaker.



(a) (b)

Figure 4 : (a) Optical pick up head, (b) head stack of hard disk drive

2.2. Mathematical model of the VCM embedded with guide pins

Figure 1 illustrates the physical structure of the VCM. The congeries of the magnet (a), the yoke (d), and the lens holder (e) forms the actuator, while the guide pins (b), the coils (c), and the CMOS sensor cover (f) are stationary parts. The current (denoted by *i*) through the coils will generate force to move the actuator along the guide pins.

Let the displacement and velocity of actuator as *d* and *v*, respectively. From [2], the S-domain model of the VCM can be seen as Figure 5. We derive physical equations in S-domain as

$$\begin{cases} \overline{U}(S) = U(S) - K_b V(S) \\ \overline{U}(S) = I(S)(LS + R) \\ F_{net}(S) = K_f I(S) - F_D(S) \\ F_{net}(S) = V(S)(MS + B) \\ V(S) = D(S)S \end{cases}$$

$$\Rightarrow \begin{cases} D(S)S = V(S) \\ V(S)S = \dfrac{-BV(S) + K_f I(S) - F_D(S)}{M} \\ I(S)S = \dfrac{-K_b V(S) - RI(S) + U(S)}{L} \end{cases}$$

Take inverse Laplace transform of above equations, the mathematical model of the VCM is described as follows:

$$\begin{cases} \dot{d} = v \\ \dot{v} = \dfrac{-Bv + K_f i - F_D}{M} \\ \dot{i} = \dfrac{-K_b v - Ri + u}{L} \end{cases}$$

where M and F_D are, respectively, the mass and the friction of the actuator, L and R are, respectively, the inductance and the resistance of the coils, K_f is the magnetic force constant, K_b is the back-emf constant, and u is the input voltage.

For simplicity, define the parameters of

$$a = \frac{-B}{M}, b = \frac{K_f}{M}, c = \frac{-1}{M}, q = \frac{-K_b}{L}, e = \frac{-R}{L}, f = \frac{1}{L}$$

and the state variables of

$$x_1 = d - d^*,\ x_2 = v,\ x_3 = i$$

The VCM model can be rewritten in the form of a state equation as

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = ax_2 + bx_3 + cF_D \\ \dot{x}_3 = qx_2 + ex_3 + fu \end{cases} \tag{1}$$

x_1 is defined as output of the system. It is an output regulation problem which falls into mismatched condition because F_D and u are in the different equations. Thus direct compensation and cancellation cannot be used here.

According to the friction model in [3], the friction F_D consists of the stiffness and the damping parts in the form of

$$F_D = \sigma_0 z + \sigma_1 \dot{z} \tag{2}$$

where z is average deflection of the bristles, σ_0 and σ_1 are the stiffness and damping coefficient corresponding to Stribeck effect, and

$$\begin{cases} \dot{z} = x_2 - \dfrac{|x_2|}{g(x_2)} z \\ \sigma_0 g(x_2) = F_C + (F_S - F_C) e^{-\left(x_2 / v_S\right)^2} \end{cases} \tag{3}$$

with F_C is the coulomb friction force, F_S is the static friction force and v_S is the Stribeck velocity.

It should be remarked that F_D does not include the viscous friction force which actually is positive for stability of VCM system. The proposed controller is intended to compensate the part of nonlinear friction, which induces stick slip oscillation, marked in figure 6

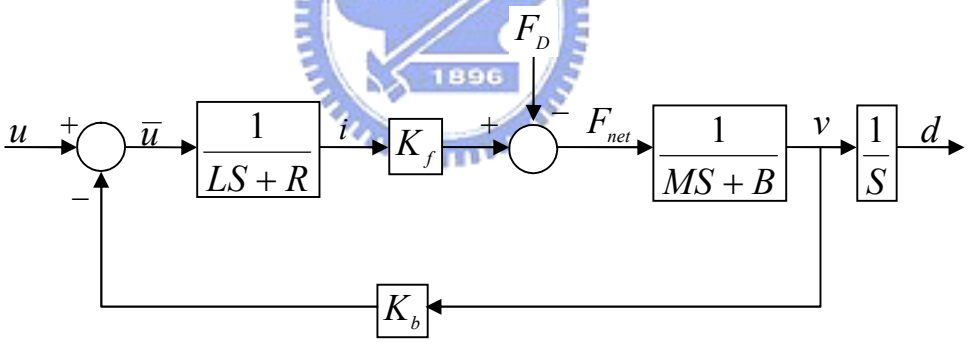


Figure 5 : S-domain model of VCM

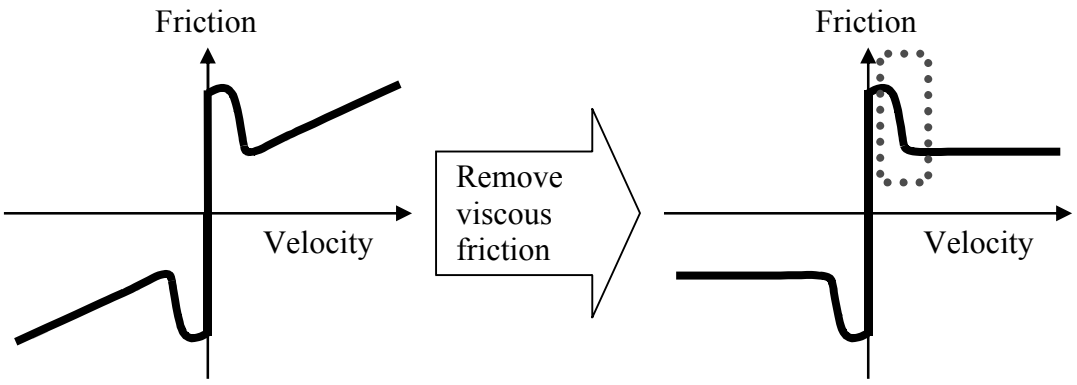


Figure 6 : The nonlinear friction induces stick slip oscillations.

Chapter 3 Proposed Control Law

Sliding mode controller is chosen to be the friction compensator due to its good reputation of friction compensation in the industry. It is also capable of changing structure of system to be a simpler one by eliminating one state variable in sliding mode. Its characteristic of variable structure enables us to create a scheme to achieve arbitrarily small steady state error while the system falls into mismatched condition.

Consider the VCM model (1)-(3), a sliding mode controller will be designed for the position control. First, choose the sliding function s as

$$s = x_2 - gx_3 - hx_1 \tag{4}$$

where g and h are control gains to be selected later such that x_1, in the sliding surface $s = 0$, will approach zero.

The next mission is to design the switching input u in (1) that drives the state variables of the system to the sliding surface $s = 0$. According to Lyapunov's theory, this can be achieved if $s\dot{s} < -\varepsilon|s|$ for $\varepsilon > 0$ and $s \neq 0$. Taking derivative of (4) and substituting (1) into it, we obtain

$$\dot{s} = (a - qg - h)x_2 + (b - eg)x_3 - fgu + cF_D \tag{5}$$

Select

$$u = \frac{(a - qg - h)x_2 + (b - eg)x_3 + \lambda^+ s}{fg} + \frac{\mu^+ \operatorname{sgn}(s)}{fg} \tag{6}$$

where sgn(s) is the sign of s, λ^+ and μ^+ are positive constants. Substituting (6) into (5) yields

$$\dot{s} = -\lambda^+ s - \mu^+ \operatorname{sgn}(s) + cF_D \tag{7}$$

which implies

$$\begin{aligned} s\dot{s} &= -\lambda^+ s^2 - \mu^+ s \operatorname{sgn}(s) + cF_D s \\ &\le -\mu^+ |s| + |cF_D s| = -(\mu^+ - |cF_D|)|s| \end{aligned}$$

Let $F_D^{\max}$ be the static friction. Thus, $|F_D| \le F_D^{\max}$ and

$$s\dot{s} \le -(\mu^+ - |c|F_D^{\max})|s| < -\varepsilon|s|$$

when selecting

$$\mu^+ = \varepsilon + |c| F_D^{\max} \tag{8}$$

This shows that u in (6) with μ^+ in (8) and any $\lambda^+ > 0$ can drives the state variables of the system to the sliding surface $s = 0$, whenever they are not in the surface. Note that λ^+ provides a freedom to adjust the approaching speed as can be seen from (7). The stability remains even if $\lambda^+ = 0$.

It should be remarked that the undesired chattering of the sliding mode control can be remedied by replacing sgn(s) in (6) with the following saturation function of

$$\text{sat}(s) = \begin{cases} 1, & \text{for } s > \beta \\ \dfrac{s}{\beta}, & \text{for } |s| \le \beta \\ -1, & \text{for } s < -\beta \end{cases} \tag{9}$$

where $\beta > 0$ represents the thickness of the boundary layer. The cost of adopting sat(s) is that the steady state error will become larger.

Now, return to deal with the selection of the control gains g and h in (4). It follows from (4) that in the sliding surface $s = 0$

$$x_3 = \frac{x_2}{g} - \frac{h}{g} x_1$$

The state equation (1) can then be reduced to a 2nd-order differential equation:

$$\ddot{x}_1 - \left(a + \frac{b}{g}\right)\dot{x}_1 + \frac{bh}{g} x_1 = cF_D \tag{10}$$

We take Laplace transform of above equation to obtain

$$X_1(S) = \frac{Sx_1(0) + \dot{x}_1(0) - \left(a + \frac{b}{g}\right)x_1(0)}{S^2 - \left(a + \frac{b}{g}\right)S + \frac{bh}{g}} + \frac{L[cF_D(t)]}{S^2 - \left(a + \frac{b}{g}\right)S + \frac{bh}{g}} \tag{11}$$

Assume that the characteristic equation of (11) has double roots of $-\lambda < 0$. The time-domain solution to (11) is then

$$x_1(t) = \alpha_1 e^{-\lambda t} + \alpha_2 t e^{-\lambda t} + \int_0^\infty cF_D(t-\tau)\tau e^{-\lambda\tau} d\tau$$

which yields the final value of x_1 as $t \to \infty$:

$$
\begin{aligned}
|x_1(\infty)| &= \left| \int_0^\infty cF_D(t-\tau)\tau e^{-\lambda\tau} d\tau \right| \\
&\le \int_0^\infty |cF_D(t-\tau)| |\tau e^{-\lambda\tau}| d\tau \\
&\le |c| F_D^{\max} \int_0^\infty \tau e^{-\lambda\tau} d\tau \\
&= |c| F_D^{\max} \left(-\frac{\tau e^{-\lambda\tau}}{\lambda} - \frac{e^{-\lambda\tau}}{\lambda^2} \right) \Bigg|_0^\infty \\
&= \frac{|c| F_D^{\max}}{\lambda^2}
\end{aligned}
\tag{12}
$$

Consequently, the steady state value $x_1(\infty)$ is bounded and can be made as small as possible by increasing λ. In a practical problem, the bound of $F_D^{\max}$ is known, so λ is calculated from (12) for a given bound of $x_1(\infty)$. After λ is determined, g and h are obtained from the following equation:

$$
S^2 - \left(a + \frac{b}{g} \right) S + \frac{bh}{g} = S^2 + 2\lambda S + \lambda^2 = 0 \tag{13}
$$

i.e., $g = -b/(2\lambda+a)$ and $h = -\lambda^2/(2\lambda+a)$.

Figure 7 shows the completed control law is composed of high gain controller, which is derived with smaller steady state error specification, and low gain controller, which is derived with larger steady state error specification. A switch is used to select low-gain or high-gain controller. Low-gain controller is selected when $|x_1|$ is larger than or equal to x_{th}, while high-gain controller is selected when $|x_1|$ is less than x_{th}. It intends to get better positioning performance without inducing large overshoot.

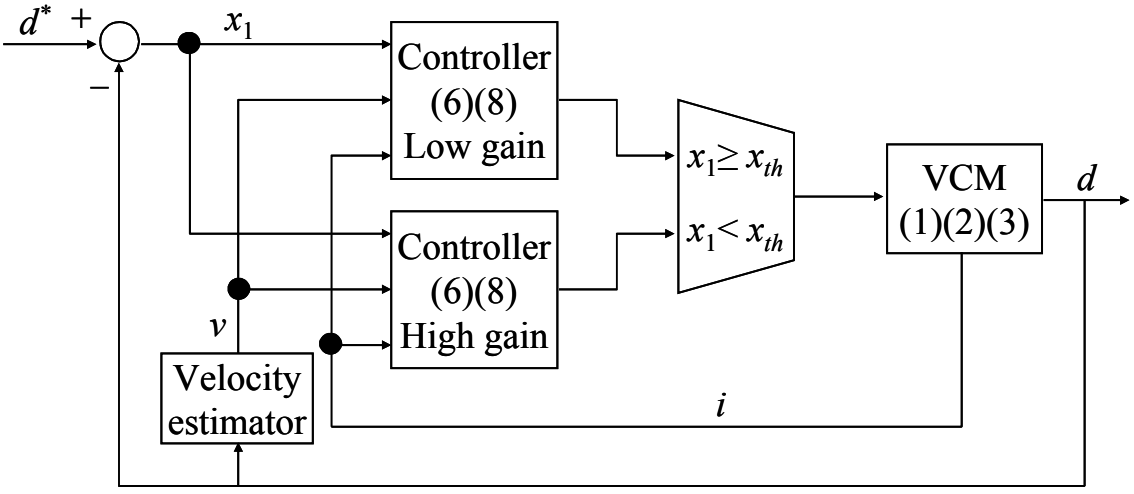


Figure 7 : Block diagram of completed proposed controller.

Chapter 4 Simulations

In this section, we use computer simulations to illustrate the effectiveness of the proposed control scheme. Consider the system (1) with a=-24, b=800, c=-1000, q=-2666.7, e=-66666.7, f=3333.3, $F_D^{\max}$=0.011. Assume the design goal is to make the steady state error less than 0.4um. According to (12) and (13), we obtain λ=5244.044, g=-0.076 and h=-2628.036. Let λ^+=0 and choose ε to have.

$$\frac{\mu^+}{fg} = -0.275 .$$

The Simulation results are shown in Figure 8. There is a steady state error ($d - d^*$) of 0.0973um, which is less than the design goal of 0.4um. This proves that the requirement of the steady state error is able to be achieved by pole placement of sliding mode state equation.

(7) is the dynamic equation of sliding function s with eigenvalue $-\lambda^+$. Approaching speed can be adjusted by tuning λ^+. As shown in Figure 9, the larger λ^+, the faster the approaching speed. Approaching speed is also affected by μ^+. As shown in Figure 10, sliding function s approaches zero faster with larger μ^+ while λ^+ is set to zero. Figure 10 also shows an interesting result that the transient response of displacement is linear while λ^+ is set to zero and the transient velocity is controlled by μ^+.

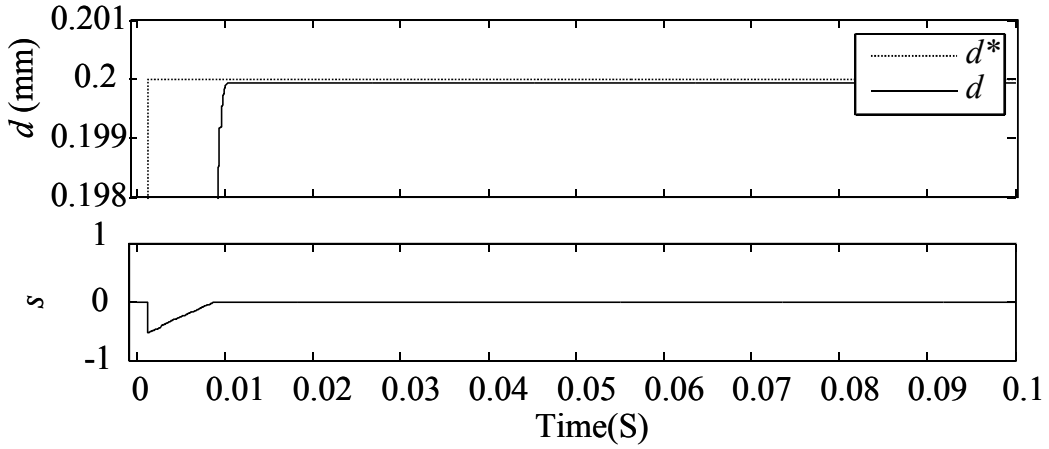


Figure 8 : Displacement steady-state response with the corresponding sliding function s.

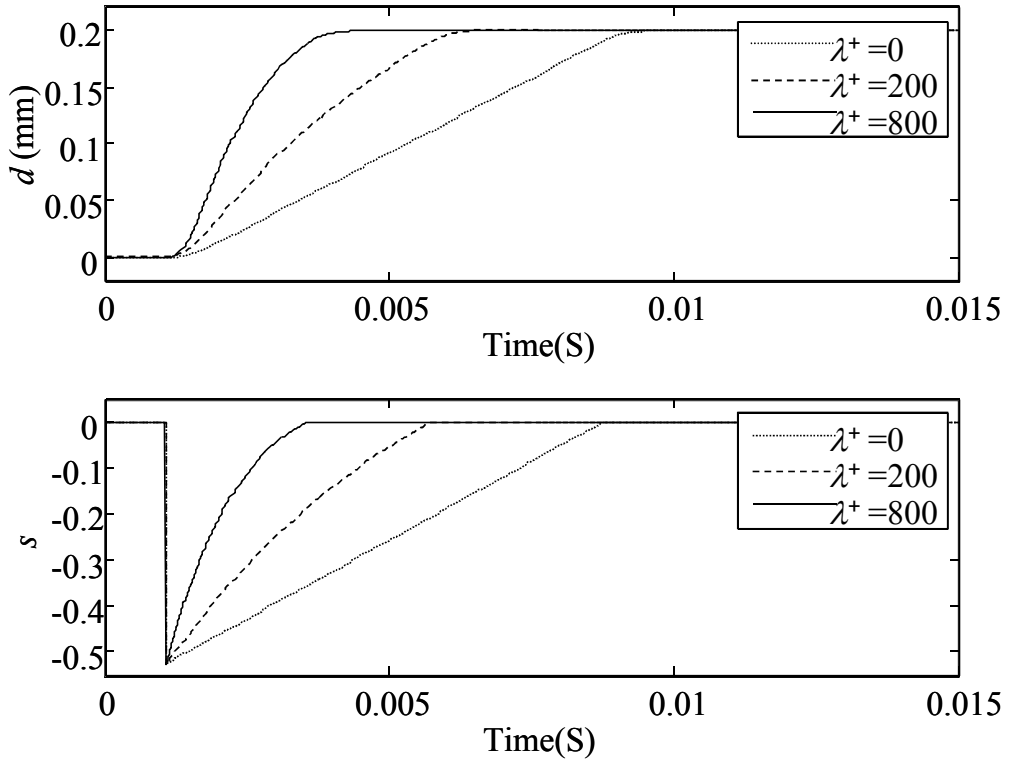


Figure 9 : Displacement and sliding function response for different value of λ^+.

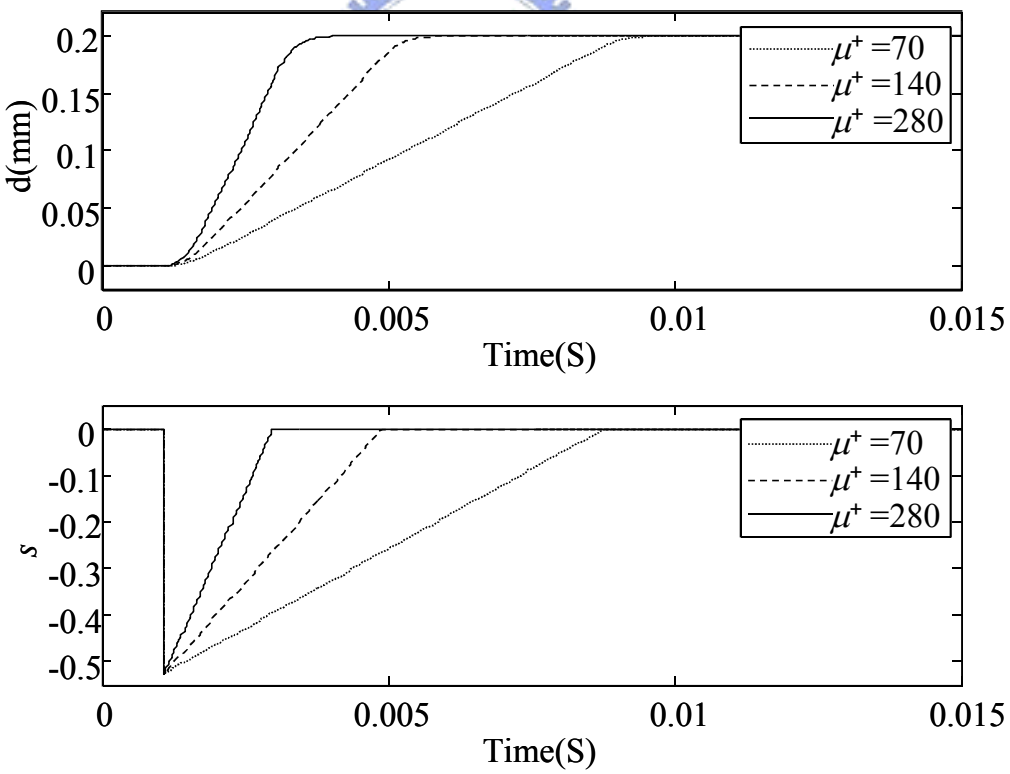


Figure 10 : Displacement and sliding function response for different value of μ^+ while λ^+ is set to zero.

The undesired chattering of the sliding mode control can be remedied by replacing sgn(s) in (6) with saturation function (9). Choose β = 0.0001 for the saturation function (9). Figure

11 shows the simulation result for adopting sign function and saturation function in the control law (6). The displacement response is very similar for adopting sign function and saturation function, as shown in Figure 11(a). But there is still a delta 2.7223×10^{-7}mm can be seen in the steady state. This proves that replacing sign function with saturation function would pay the cost of enlarging steady-state error. Figure 11(c) and Figure 11(d) demonstrate that saturation function can totally remedy the undesired chattering while large enough β is chosen.

A classic PI control law, as shown in Figure 12, is compared with the proposed controller to show the improvement. It is apparent from Figure 13(a) that there are stick slip oscillations in the steady-state of the classic PI controller. On the contrary, the proposed sliding mode controller doesn't induce any stick slip as shown in Figure 13(c). Besides the ability to compensate for the nonlinear friction force of the VCM, the proposed controller also has faster transient response, which can be seen from Figure 13(b) and 13(d).

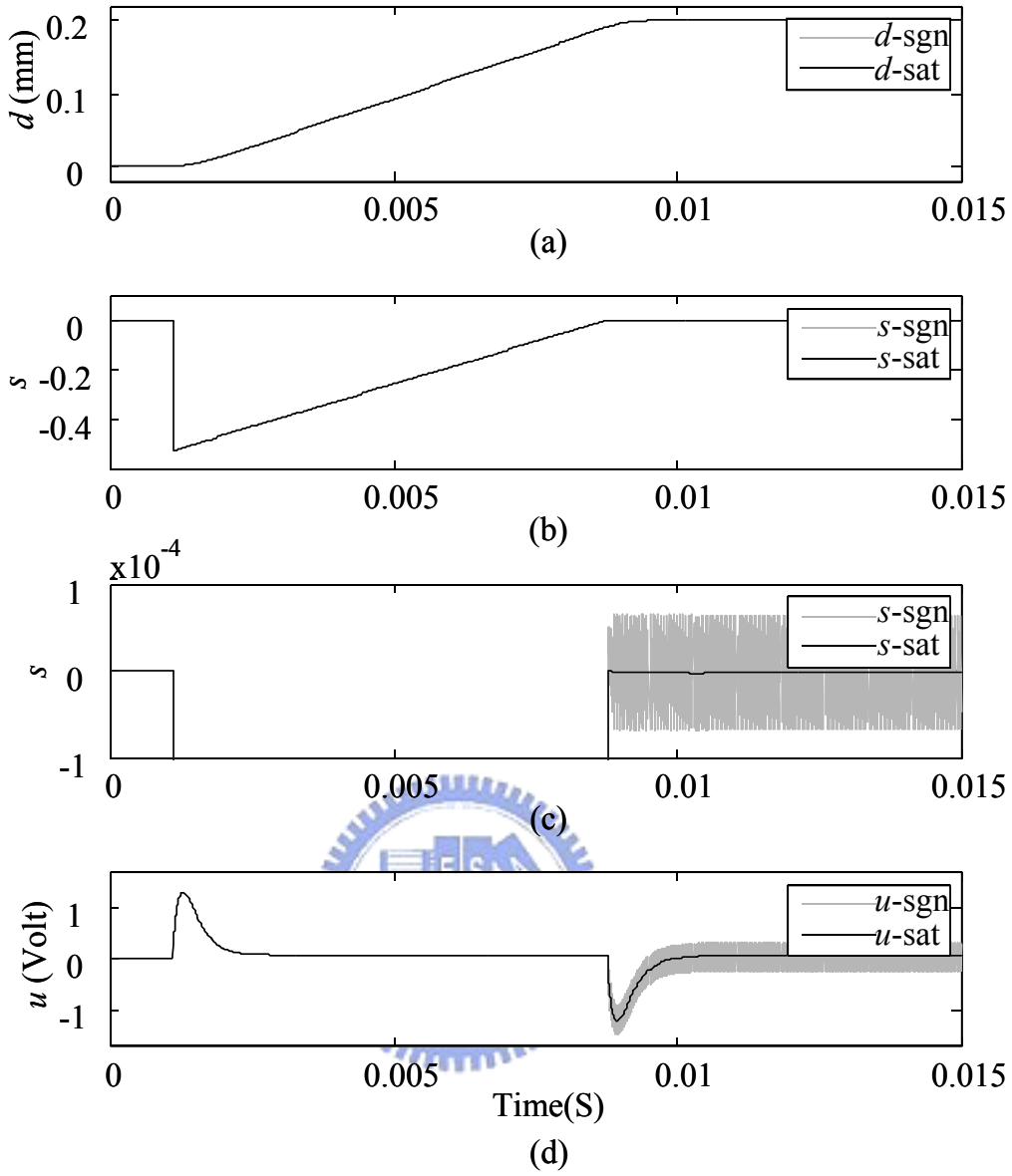


Figure 11 : The response of adopting sign function and saturation function in control law (6). (a) Displacement, (b) sliding function *s* in large scale, (c) sliding function *s* in small scale and (d) control law *u*.

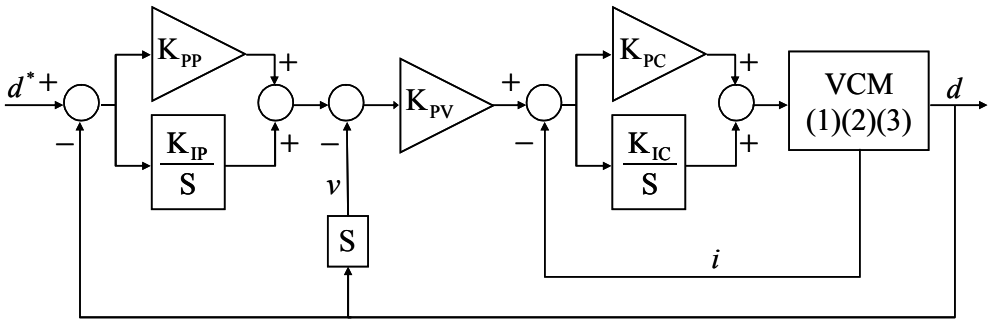


Figure 12 : Classic control law with PI controller on the position loop.

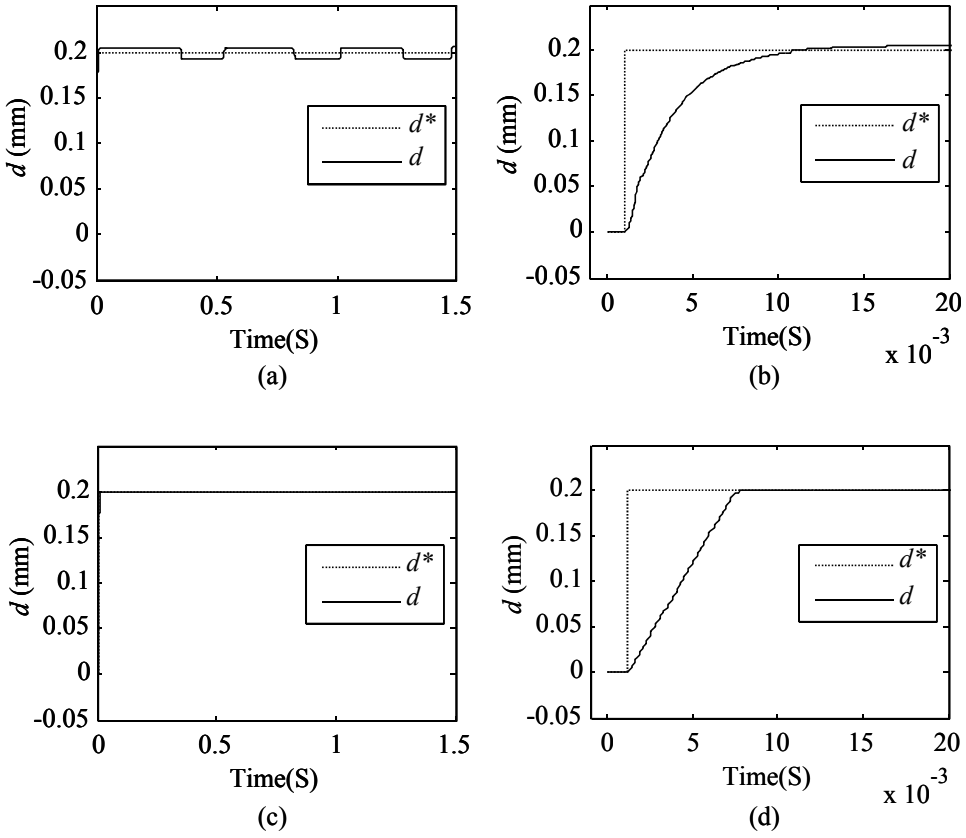


Figure 13 : (a) Steady response with stick slip and (b) transient response for classic PI controller. (c) Steady response without stick slip and (d) transient response for sliding mode controller.

Chapter 5 Experimental environment implementation

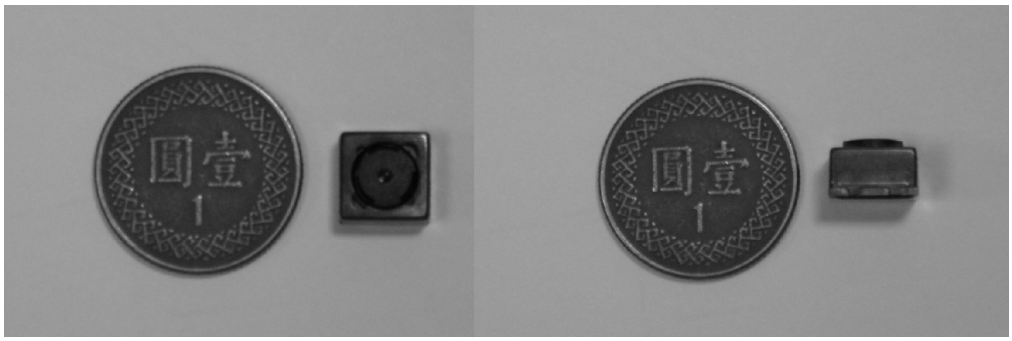
PC motherboard is chosen to be the calculation unit instead of embedded system due to its high calculation power and low cost. Another important consideration is the CPU for PC platform supports floating point arithmetic operation. Taking AMD Sempron 2600+ CPU as an example, it can complete 1000 double-type multiplication-and-add calculations within 12.02us in DOS system under our test. This calculation power is able to support advanced control algorithm. Additionally, algorithm migration from MATLAB m file to C code is also smooth because the syntax of m file and C are similar. The best part is that double-type can be used for all variables in C code while control algorithm is developed in the PC platform. Although the double type variables are also supported by the C compiler for ARM and 8051 platform, but the calculation delay for the arithmetic operations will be too long to support high enough sampling rate for advanced control algorithm. Because ARM and 8051 processor does support floating point in hardware, the double-type arithmetic operation is done by software emulation.

Our purpose is to build up an experimental environment, in which the proposed control algorithm can be validated effectively before it is implemented on FPGA. Migrating control algorithm from MATLAB to FPGA, in our experience, would take 20 times, or even longer, the time of migrating from MATLAB to C code. An experienced engineer with good FPGA developing skill can benefit from adopting this platform in his design flow by saving the iteration time of changing the control algorithm in validating phase, not to mention those students who are new to implementing algorithm on FPGA. Appendix A presents the control algorithm design flow, in which a PC-based validating platform is included.

The PC-based experimental environment is composed of a motherboard and an FPGA board, as shown in Figure 14(a). Standard PATA interface is selected to connect these two parts because of its simplicity and convenience (most motherboards support, at least, one PATA port). “Register transfer protocol” of PATA specification [7] is adopted for data transfer from/to the FPGA board. The VCM used for experiment is shown in Figure 14(b). Its

size is smaller than one dollar coin. Below we brief the environment implementation. Appendix B copies the needed information from PATA specification for convenience.

(a)



(b)

Figure 14 : (a) Picture of PC-based experimental environment which includes a motherboard and an FPGA board. The motherboard and FPGA board are connected by ATA cable. (b) The VCM used for the experiment.

5.1. Hardware implementation

There are 2 parts in the hardware implementation, one is the board design and the other is the FPGA design. The block diagram of the complete hardware is shown in Figure 2.

5.1.1. Board design

Board design takes care of two things, one is yielding the discrete data for current and displacement feedback signals and the other one is converting the controller's PWM signal to physical voltage to drive VCM.

The current signal and displacement signal are amplified by difference amplifiers first. Anti-aliasing filters then is used to attenuated the high-frequency noise, which may induce low frequency alias after sampling, of the output signals of the difference amplifiers. The anti-aliasing filter is composed of resistor and capacitor (RC filter). The output signals of the anti-aliasing filter then sampled by analog-to-digital converter (ADC). The discrete signals of current and displacement finally input to FPGA chip. It should be remarked that ADC is very sensitive to noise, we strongly recommend to use a dedicate voltage regulator for ADC. Do not share its power source with other components.

The digital controller's output is PWM signal. A full bridge is used to convert the PWM signal to physical voltage to drive the VCM. A well known alternative solution for the power converter is power operational amplifier (power OP). Comparing to power OP, full bridge does not only support higher power conversion efficiency, but also has smaller die size when going to ASIC implementation. But using full bridge as power converter raises the challenge of implementing current signal sensing, because current signal is a switching signal. Synchronous sampling scheme [4] is adopted to derive better current sensing response and will be explained later.

5.1.2. FPGA design

FPGA design is composed of ATA IO module, PWM module and 2 discrete IIR filter modules. The implementation details are described in the following sub-sections.

5.1.2.1. ATA IO module

ATA IO module handles the ATA register transfer protocols, which enables control algorithm to access the registers, such as parameters of IIR filter, voltage command, discrete current feedback and discrete position feedback.

The IO address of on-board primary PATA port is standardized as 0x1F0~0x1F7. Data width of IO port 0x1F0 is 16-bit, others are 8-bit. This limited register space can be expanded to 256 × 16-bit, as shown in Table 1, by defining 0x1F1 and 0x1F0 as index register and data port respectively. The protocol of accessing the expanded register space is 1) write target address to index register and 2) read/write data port. Take “read/write from/to Reg5[15:0]” as an example, 1) write 5 to 0x1F1 and 2) read/write 0x1F0.

Table 1 : Expand register space on PATA bus

IO port	CS0#	DA[2:0]	Index[7:0]	Register
0x1F1	0	1	X	Index[7:0]
0x1F0	0	0	0	Reg0[15:0]
0x1F0	0	0	1	Reg1[15:0]
0x1F0	0	0	2	Reg2[15:0]
⋮	⋮	⋮	⋮	⋮
0x1F0	0	0	255	Reg255[15:0]

Control algorithm, executed on the motherboard, reads position and current feedback registers of FPGA board through PATA interface. The output data of FPGA board must be synchronized by IOR# signal, or transition-state data will possibly be read by control algorithm. The transition-data, as shown in Figure 15, is actually unpredictable, neither stable value 123 nor 456. Taking the transition-data as current or position feedback would fail any control algorithm. In the experimental environment, position and current feedback register is updated according to the clock signal, which is asynchronous to IOR# signal, on the FPGA board. Figure 16 shows the synchronization circuit and its timing, which guarantees the data

integrity of the read data. The LTCH signal is synchronous to IOR# and is used to latch the output data OD[15:0] which is sent to the motherboard.

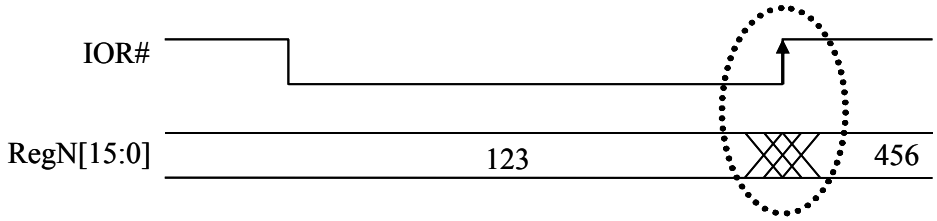


Figure 15 : Transition-state data is fetched by register read protocol of motherboard's PATA port.

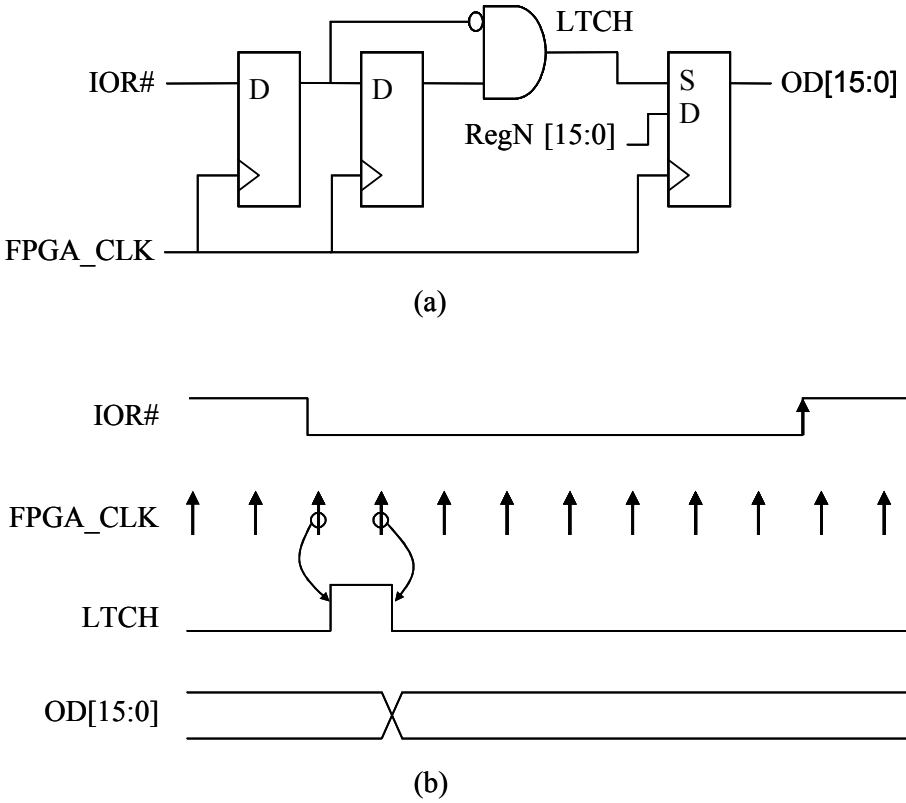


Figure 16 : (a)Circuit to yield LTCH which is synchronous to IOR#. (b)Output data OD[15:0]'s integrity is guaranteed by LTCH.

5.1.2.2. PWM module

PWM module yields PWM+ and PWM- signals with unipolar voltage switching scheme [15] to control full bridge to drive VCM. The duty of PWM+ and PWM- signals implies the level of physical voltage command as

Voltage command = [(Duty of PWM+) – (Duty of PWM-)] x (Voltage source of full bridge)

Assume the voltage source for full bridge is 3.3V, the maximum and minimum voltage command then are +3.3V and -3.3V. Figure 17 shows the electrical model of the VCM with the model of full bridge. The back-emf of the VCM is ignored here for convenience. Figure 18 shows the PWM+ and PWM- modulation with the voltage and current response for the model shown in Figure 17. It should be remarked that the current response is a switching signal. That is why the synchronous sampling is needed.

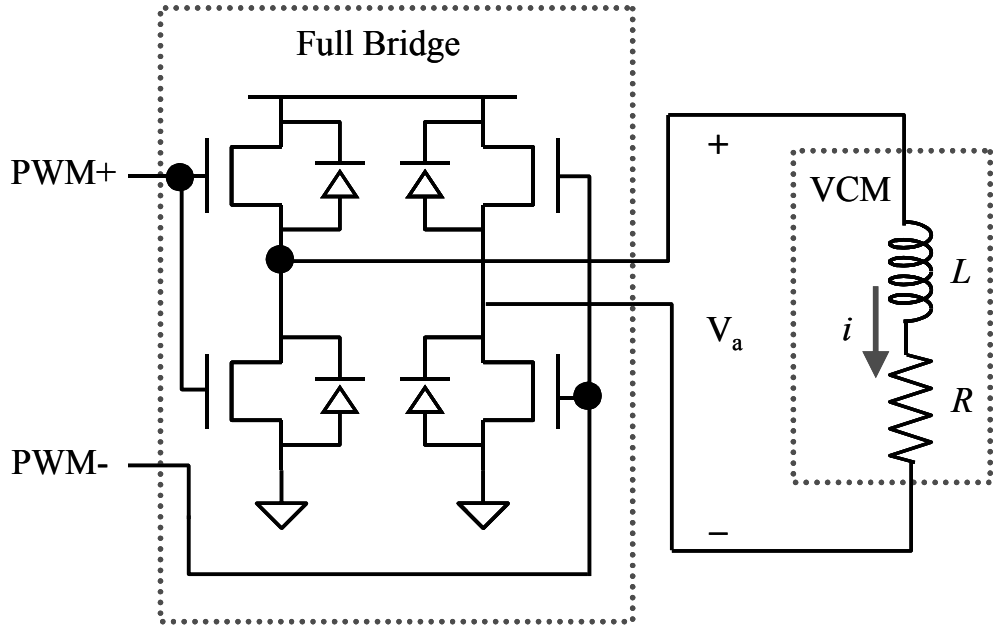


Figure 17 : Full bridge circuit and electrical model of VCM.

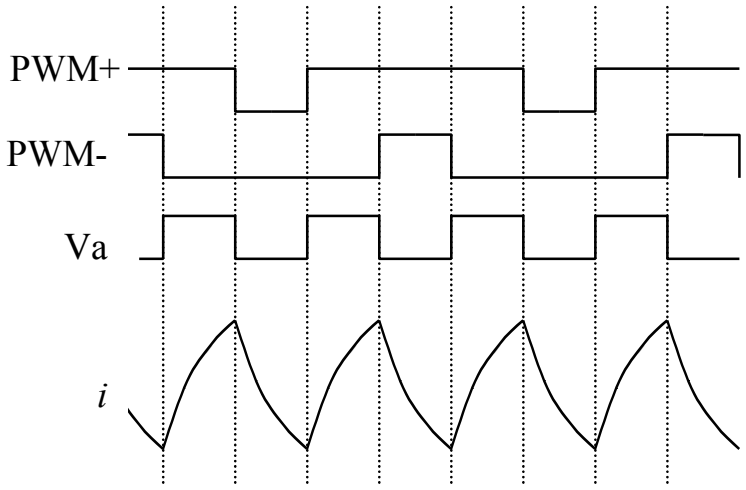


Figure 18 : PWM with unipolar voltage switching and the corresponding voltage and current response.

5.1.2.3. IIR module

IIR module is a second-order discrete Butterworth filter which is used for attenuating the noise of discrete current and displacement feedback signals. Generally, the cut-off frequency of the discrete filter is a dominant pole comparing to that of anti-aliasing filter. So that the anti-aliasing filter can be ignored for convenience when doing stability analysis.

Figure 19 shows the simulation model on Simulink, as well the design steps can also be seen on it. First step is deriving the Z-domain of the filter. Then convert the Z-domain to "direct form 1" realization form with fix-point operation. Algorithm of the IIR filter then can be derived. FPGA implementation by using hardware description language (HDL), such as Verilog and VHDL, then can be proceeded. It is a good way to validate the correctness of the algorithm by using behavior coding before implementing synthesizable coding (using state machine). Appendix C presents the Verilog behavior coding of the second order Bufferworth filter for reference. Simulink supports co-simulation with Modelsim. It allows designer to validate the FPGA implementation by comparing the output of HDL model with the output of realization model and Z-domain model. Figure 20 shows the simulation results for HDL model, realization model and Z-domain model of the filter. The result of Verilog model matches that of Z-domain model. It implies the Verilog implementation is correct and can be synthesized to FPGA chip.

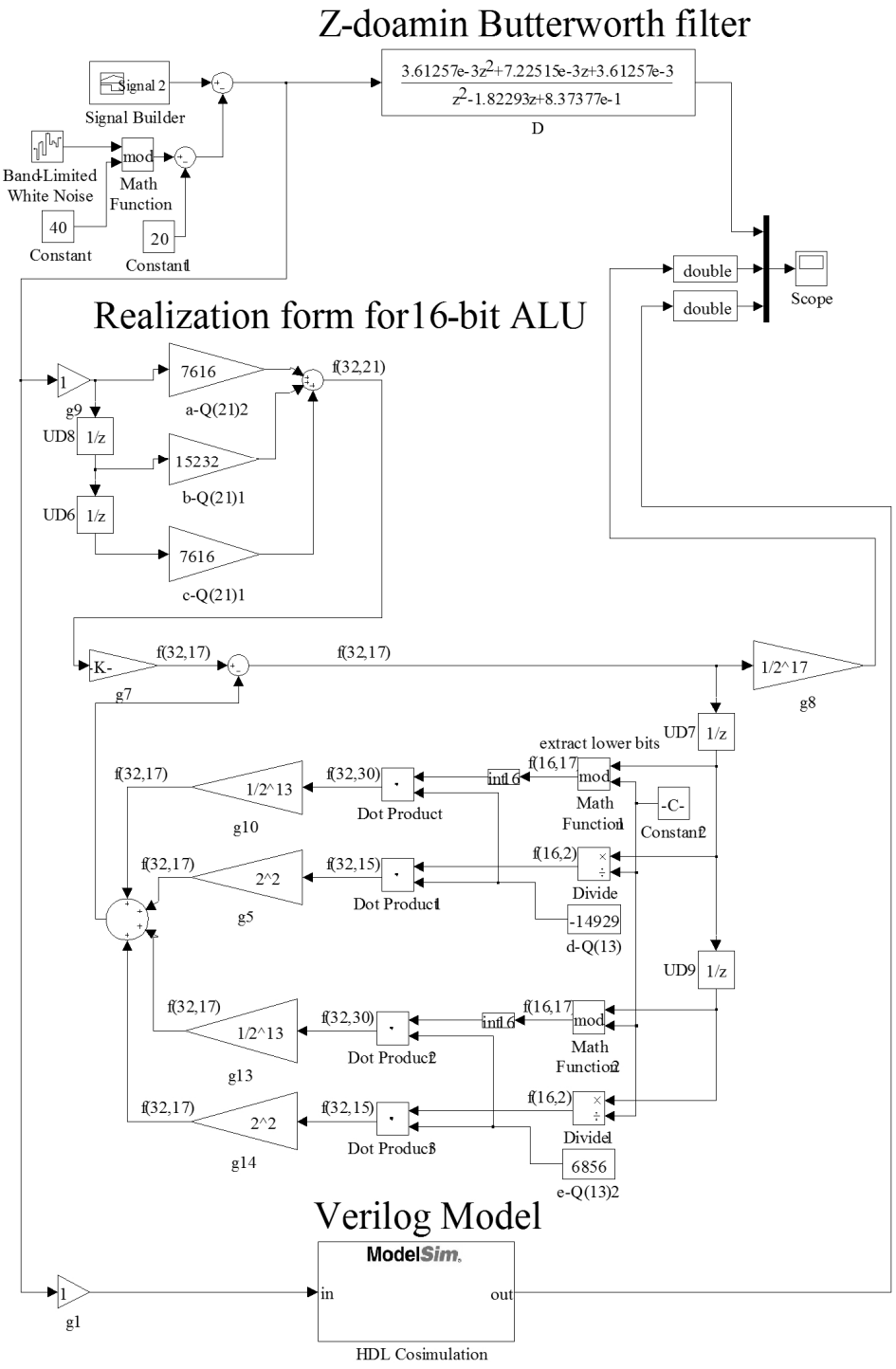


Figure 19 : Second-order Butterworth filter simulation model, including Z-domain model, realization form model and Verilog model.

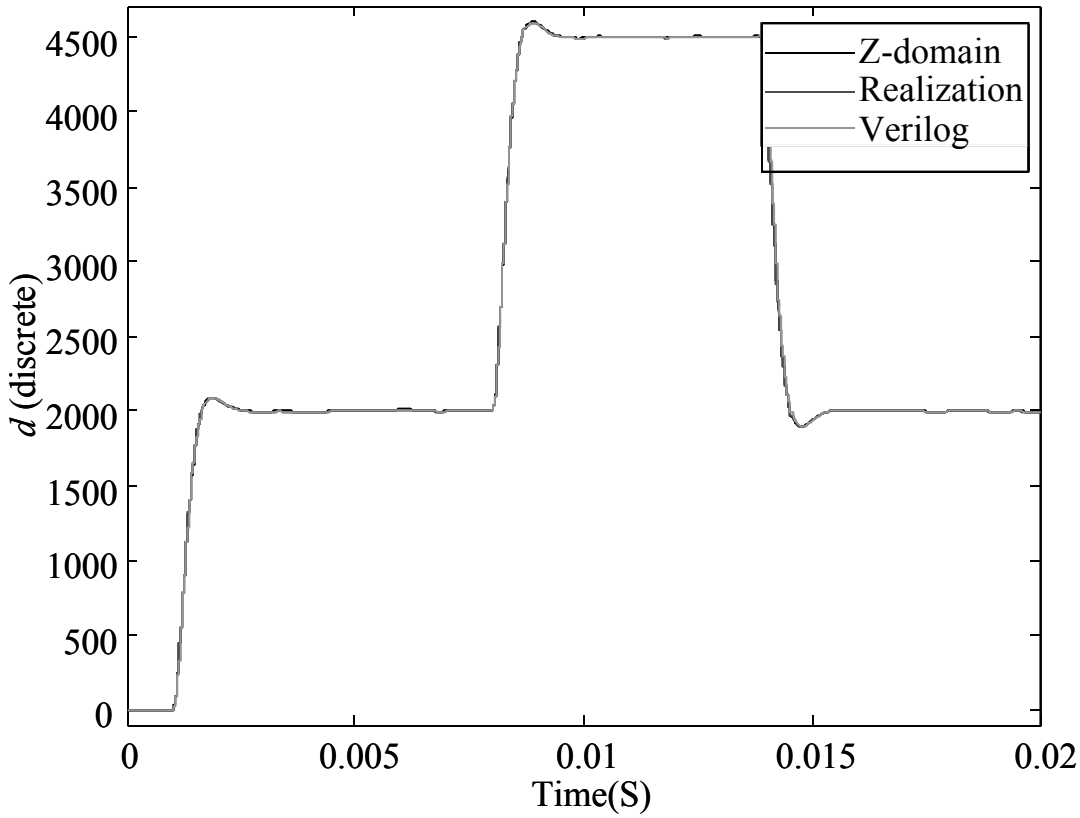


Figure 20 : The simulation results for Z-domain, realization form and Verilog models.

5.1.3. Synchronous current signal sampling

As we can see in the Figure 18, the steady-state current response of the VCM is a triangular waveform when full bridge is adopted as power converter. The discrete current feedback will not be a constant value while ADC arbitrarily samples the current signal. One alternative to eliminate the ripple current is increasing the switching frequency of PWM signals, but it will also increase the switching loss of full bridge and the operational frequency of PWM module. Adopting synchronous sampling scheme can eliminate the effect of current ripple without paying the cost of increasing frequency of PWM signals.

Synchronous sampling is having ADC's sampling the analog current signal be synchronous to PWM signal, as shown in Figure 21. The sampling of ADC is triggered by the SAMPLE signal which is synchronous to triangular TRI signal. The TRI signal is also used to yield PWM+ and PWM- signal. This guarantee the synchronization between SAMPLE and

PWM+/PWM- signals. By referring to the current response in Figure 18, average value of the current ripple is sampled and constant discrete current feedback can be derived in steady-state.

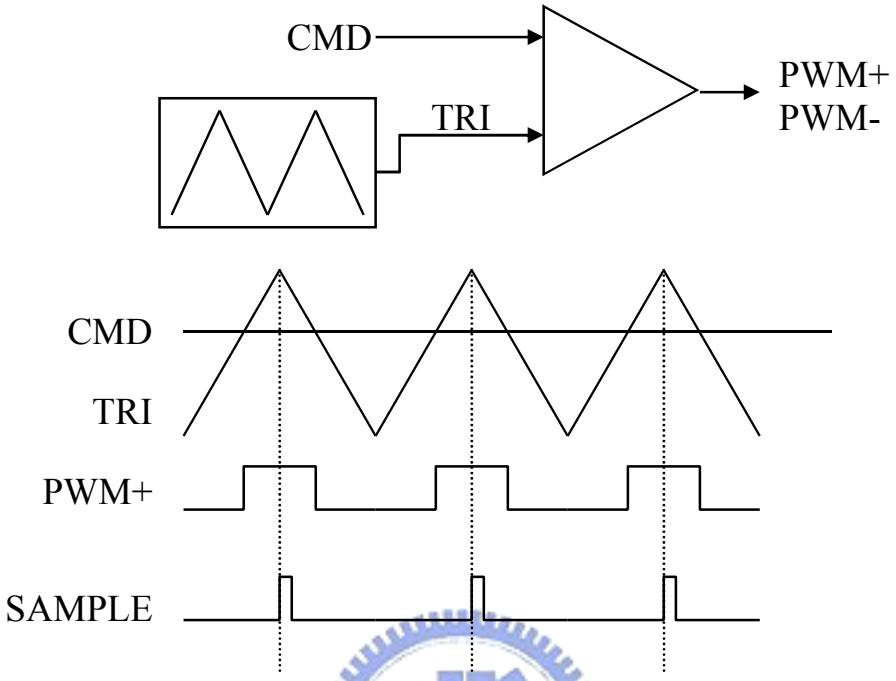


Figure 21 : The SAMPLE signal to ADC is synchronous to PWM signals.

5.2. Software implementation

Control algorithm requires consistent sampling time according to the discrete control theory. Multi-tasking operating system (OS), such as Windows and Linux, is not suitable for controller implementation. We choose DOS as development platform because it is single-tasking OS. Some tips for the software implementation of the control algorithm are shown below.

Tip1 : Two ways to control the sampling time of control algorithm in PC environment. One is to use hardware interrupt IRQ9, which is triggered by 8253. Set sampling time by programming the registers of 8253 and put algorithm on the interrupt service routine (ISR) which is corresponding to IRQ9. The other way is generating a periodic signal with specified period by FPGA and having algorithm poll the signal through a register bit. We choose the later one for our implementation and the one bit flag of the periodic signal (FPS) is merged

into a 16-bit register with displacement feedback, as shown below. Control algorithm is executed once when FPS's toggling from 0 to 1 is detected.

15	14 12	11 0

[15] : flag of periodic signal (FPS)
[14:12] : reserved
[11: 0] : discrete displacement feedback

Tip2 : Hardware interrupt of PC platform can be blocked by assembly instruction "cli" and released by "sti". All hardware interrupts, such as system timer and keyboard, must be blocked, or consistent sampling time required by the control algorithm may not be maintained. And the hardware interrupt blocking must be released at the end of the control algorithm, or the keyboard will stop working.

Tip3 : Only these motherboards, which do not implement non-mask interrupt in DOS system, are suitable for control system development. A simple verification is writing IO port 0x1F1, with arbitrary data, in an infinite loop following the assembly instruction "cli". It is qualified if IOW# signal is periodic without any interruption. We saw some motherboards implement periodic non-mask interrupt and induce large pulse of IOW# under the test, as shown in Figure 22. There motherboards are not suitable for control algorithm implementation.

Tip4 : Don't access keyboard nor print out message to screen while algorithm is running, because these IO operations result in significant time delay.

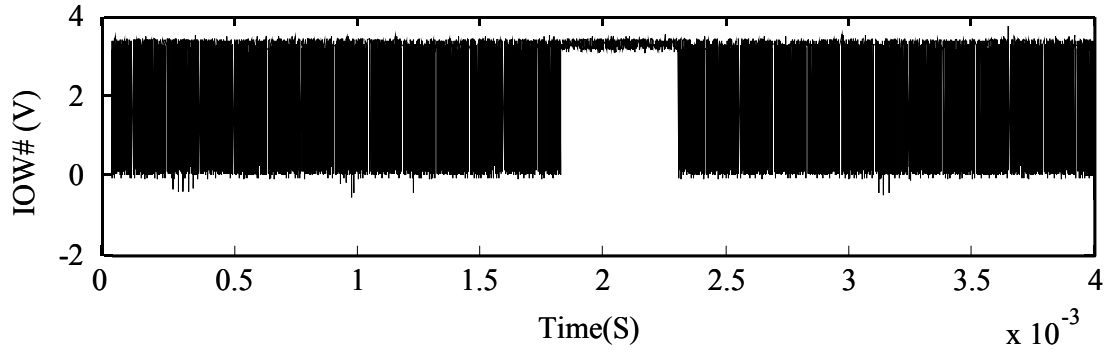


Figure 22 : Unexpected large pulse of IOW# under the test of Tip3.

Chapter 6 Experimental results

The controller performance was studied experimentally. All experiments were performed with a product-like VCM, as shown in Figure 14(b). Control law (6) and (8) are adopted for the switching controller, as shown in Figure 7. And the tuning method introduced in [14] is used to tune the classic PI controller. The experimental setup is shown in Fig 2. The experimental system consists of motherboard and FPGA board, which includes VCM, full bridge as power converter, current sensing circuit, position sensing circuit and FPGA. The current feedback and position feedback is stored in the registers of the FPGA. Control algorithm is executed on the motherboard. It reads the position feedback and current feedback from the FPGA through PATA cable, and sends the digital voltage command to the FPGA. The digital voltage command is converted to PWM with unipolar voltage switching [15] for full bridge, which drives the VCM.

6.1. Adopt sign function in control law (6)

Figure 23(a) and 23(b) shows that the classic PI controller yields stick slip oscillations and proposed sliding mode controller doesn't. Stick slip oscillations would result in significant image-shaking and is not allowed in the camera application. Figure 23(d) demonstrates the settling time of the sliding mode controller is smaller than the classic PI controller's and it is less than 10ms. It implies that it is able to support the advanced AF algorithm capable of 60-frame rate. It should be remarked that Figure 23(c) is noisier than Figure 23(a) due to the system noise induced by the chattering of sliding mode control.

Repeatability is also a critical specification for the VCM. In camera, AF algorithm detects the clearness of image in multiple positions of the full optical stroke, then move actuator to the position with clearest image. Poor repeatability would degrade AF performance because the actuator would stay in a position different from the one the AF algorithm expects. In the repeatability experiment, a laser displacement meter (LDM) measures the physical position of

the actuator. Figure 24 shows the repeatability, the delta between cursor 1 and cursor 2, is 4um, which is less than the nominal requirement 10um.

Experiment of moving the actuator in horizontal direction is also performed. Figure 25 shows there is no stick slip oscillations in steady response and the settling time is still less than 10ms. Figure 26 shows the repeatability is 6um, which is still less than the nominal requirement 10um. Comparing to the foregoing results, the repeatability in horizontal direction is slightly poorer. The reason is that unbalanced friction force is more largely applied on the two guide pins in this horizontal orientation.

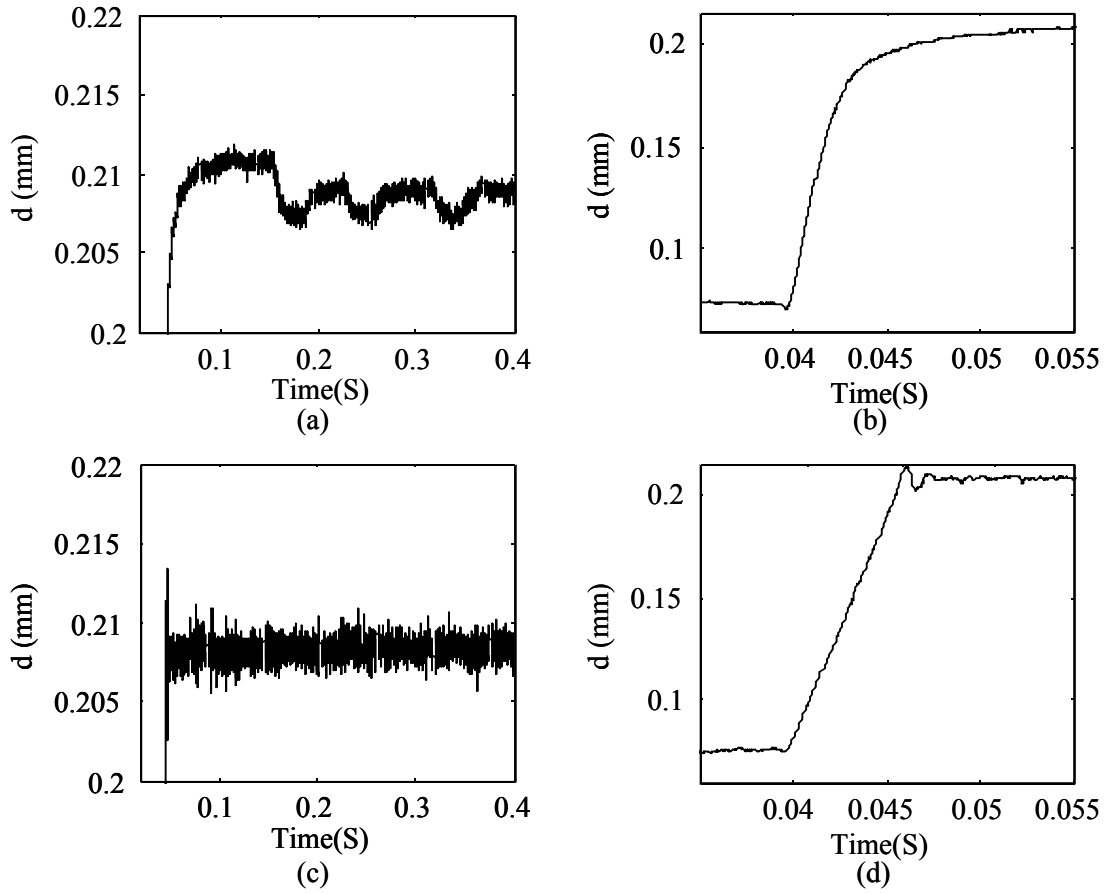


Figure 23 : (a) Steady response with stick slip and (b) transient response for classic PI controller. (c) Steady response without stick slip and (d) transient response for sliding mode controller.

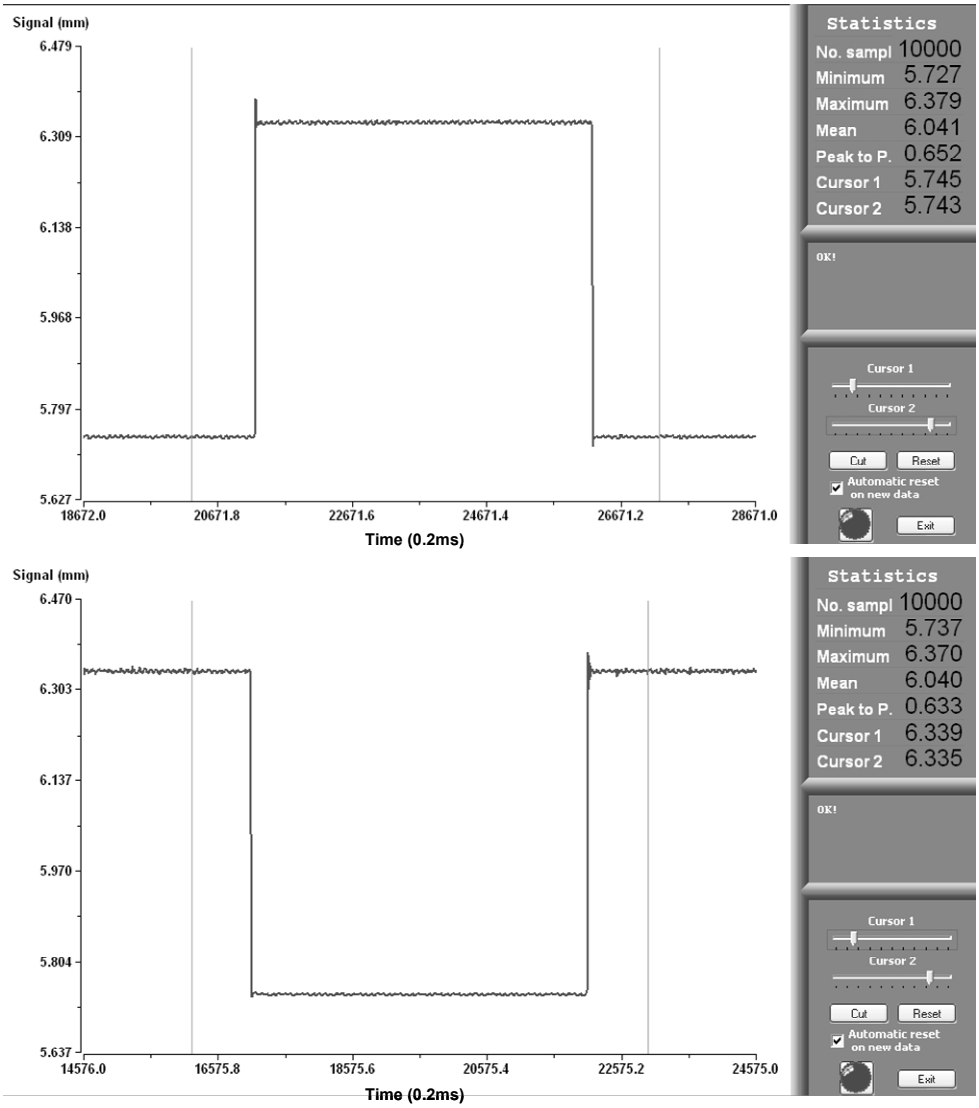


Figure 24 : Repeatability is 2um and 4um for high and low position respectively in vertical direction.

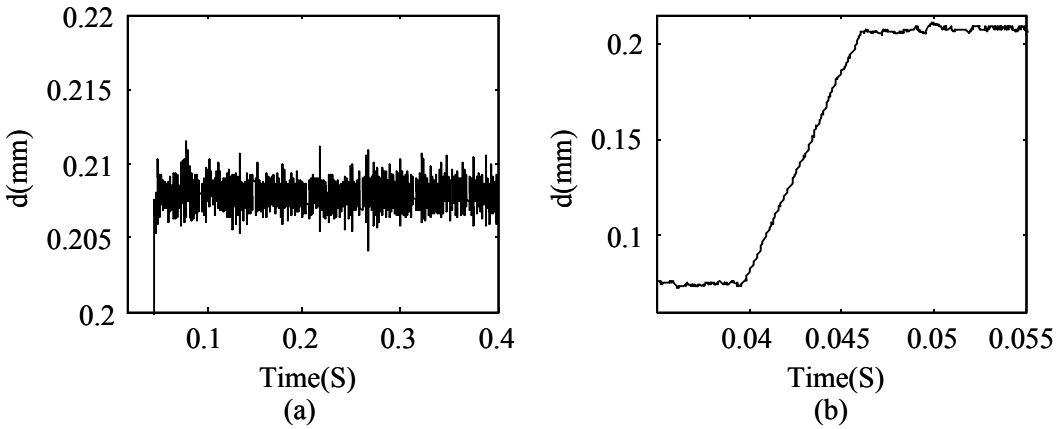


Figure 25 : Actuator moves in horizontal direction. (a) Steady response without stick slip and (b) settling time is less than 10ms.

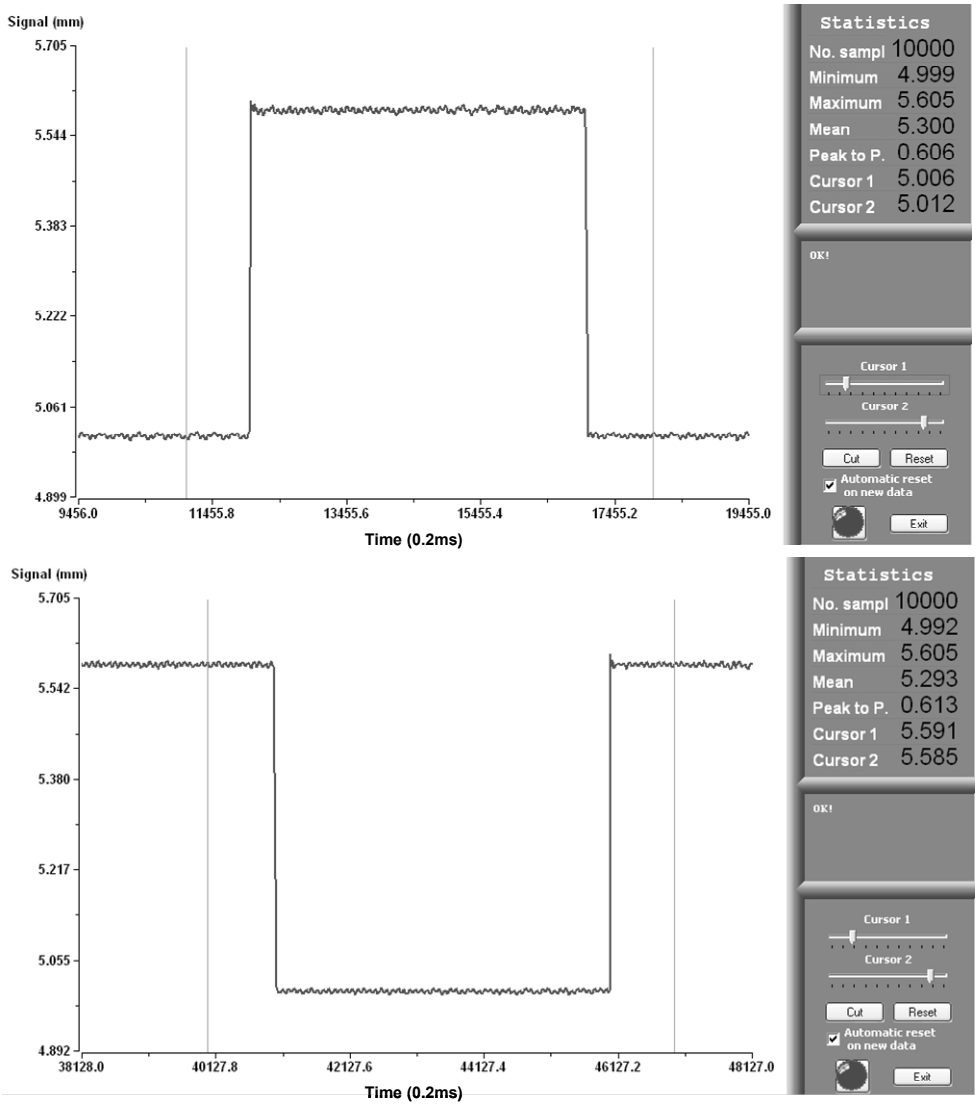


Figure 26 : Repeatability is 6um for both high and low position in horizontal direction.

6.2. Replacing sign function in control law (6) with saturation function

The undesired chattering can be remedied by replacing the sign function of control law (6) with the saturation function (9), as shown in Figure 27. Figure 27 (b) is the current *i* response hile sign function is adopted in (6), and Figure 27 (d) is the current *i* response while saturation function is adopted in (6). It is easy to be seen that the Figure 27 (b) is much noisier. This proves that the undesired chattering avoided by adopting saturation instead of sign function in control low. Besides, the repeatability is also improved while the chattering is solved, as shown in Figure 28 and Figure 29. The repeatability is reduced to 1um, which quit better than nominal requirement of 10um.

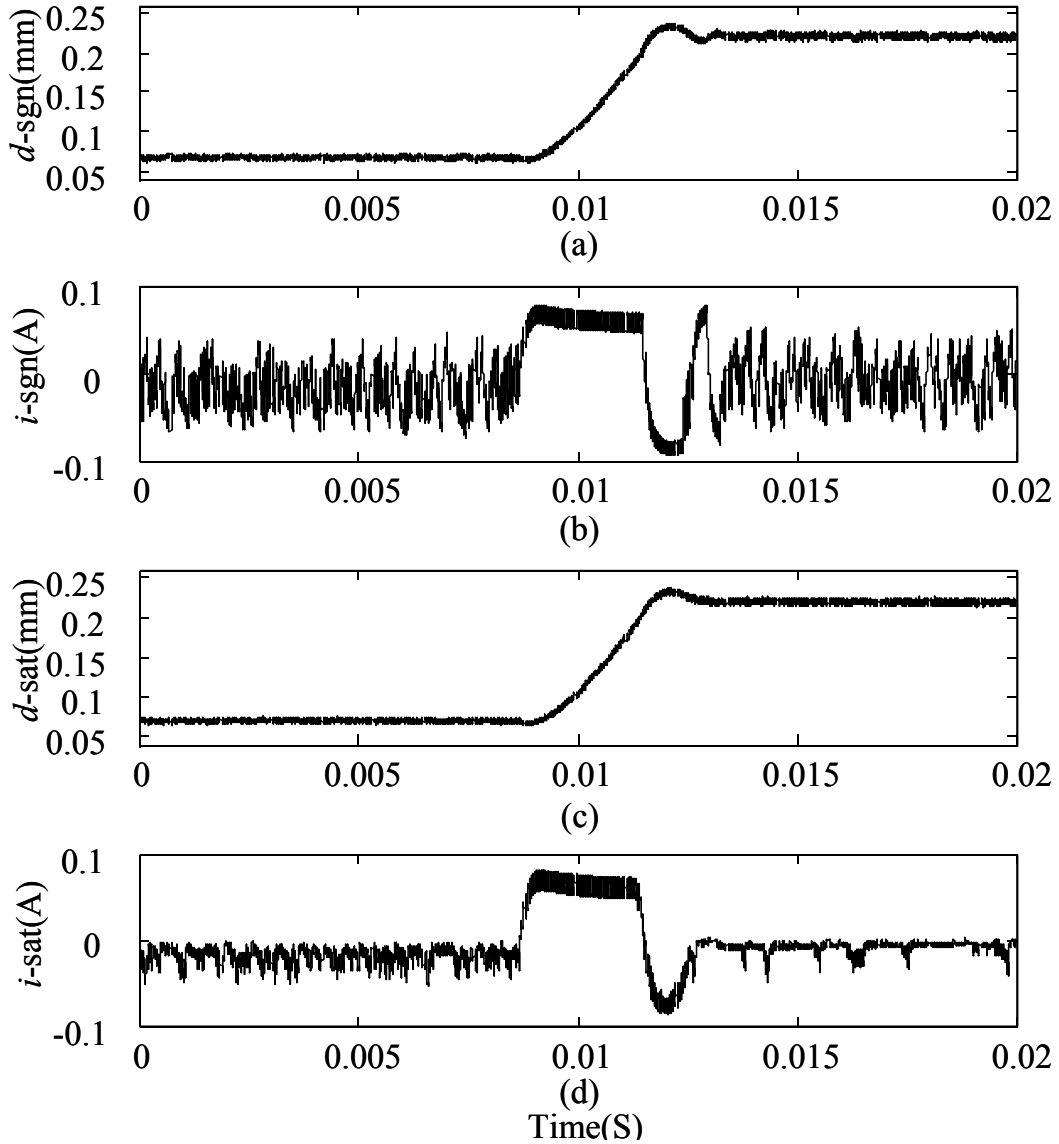


Figure 27 : (a) and (b) are displacement and current response with sign function in control law(6), (c) and (d) are displacement and current response with saturation function in control law (6).

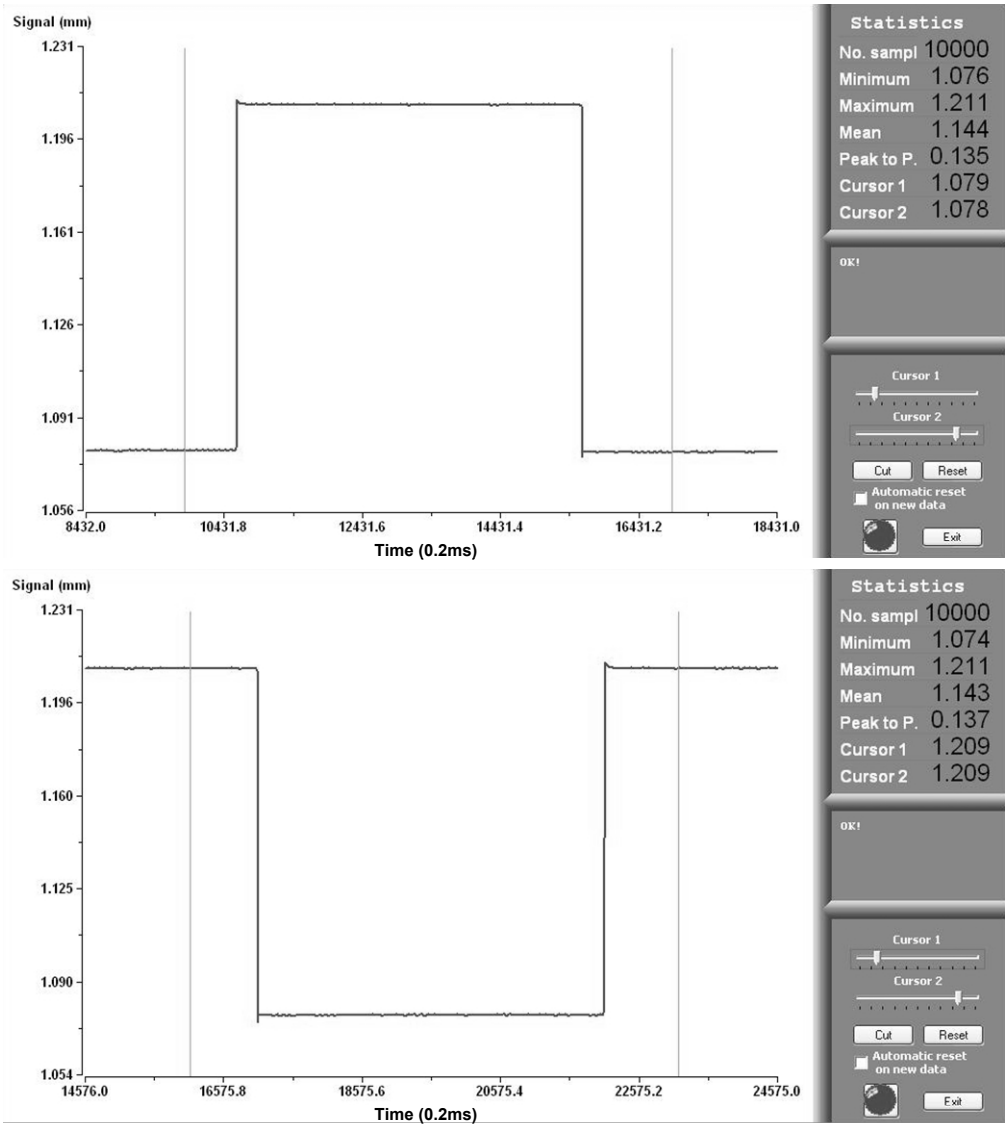


Figure 28 : Repeatability is 1um in vertical direction while saturation function is adopted.

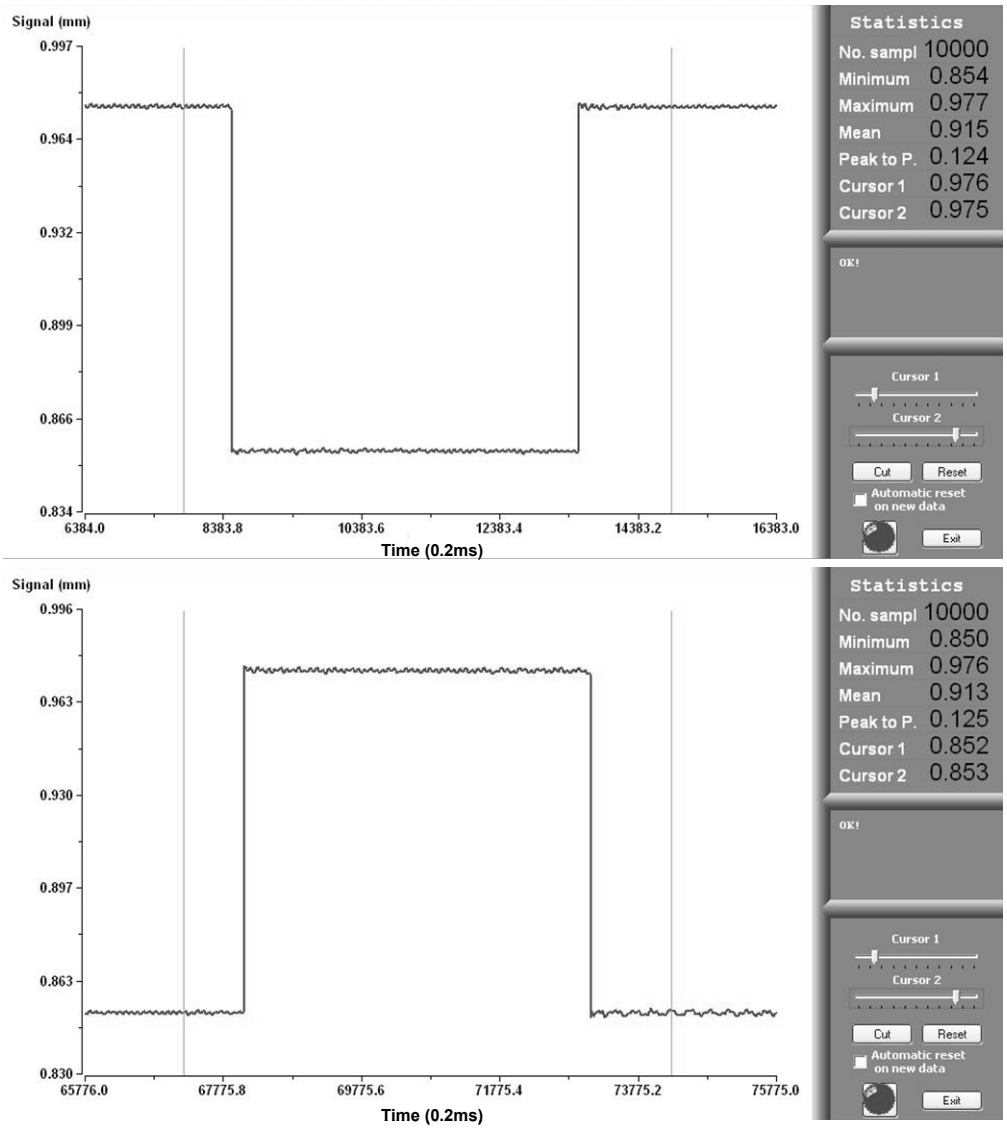


Figure 29 : Repeatability is 1um in horizontal direction while saturation function is adopted.

Chapter 7 Conclusions

In this thesis, a sliding mode controller is proposed to compensate for the nonlinear friction force of the mini VCM which is applied to CCM. Stick slip oscillation is avoided and steady state error can be designed in arbitrarily small by pole placement of the sliding mode state equation. The effectiveness of the controller is firstly proved by the simulation results. The experimental results then demonstrate that the transient response is less than 10ms, no stick slip limit cycle oscillation occurs in steady response, and repeatability performance is also satisfactory. Consequently, the proposed control scheme works well and is reliable.

A PC-based experimental environment is also introduced. It is not only cost effective but also of high calculation power. Control algorithm validation can be more efficient by using it.

References

[1] B. Bona and M. Indri, "Friction compensation in robotics: an overview", Proc. of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005, Seville, Spain, December 12-15, 2005.

[2] B. C. Kuo and F. Golnaraghi, Automatic Control Systems, 8th Edition, John Wiley & Sons, Inc.

[3] C. C. de Wit, H. Olsson, K. Åström, and P. Lischinsky, "A new model for control of systems with friction," IEEE Trans. on Automatic Control, vol. 40, no. 3, pp. 419–425, 1995.

[4] J. Chen, A. Prodić, R. W. Erickson and D. Maksimović, "Predictive digital current programmed control," IEEE Trans. on Power Electronics, vol. 18, no. 1, pp. 411-419, January 2003.

[5] B. C. Baker, "Anti-aliasing, analog filters for data acquisition systems", AN699, Microchip Technology, Inc, 1999.

[6] J. J. E. Slotine and W. Li, Applied Nonlinear Control, Pearson Education Taiwan Ltd, 1991.

[7] Information Technology -AT Attachment with Packet Interface - 5(ATA/ATAPI-5).

[8] H. C. Yu, T. Y. Lee, S. J. Wang, M. L. Lai, J. J. Ju, D. R. Huang and S. K. Lin (2005), "Design of a voice coil motor used in the focusing system of a digital video camera," IEEE Trans. on Magnetics, vol. 41, no. 10, pp. 3979-3981, 2005.

[9] H. C. Yu, T. Y. Lee, S. K. Lin, S. J. Wang, M. L. Lai, J. J. Ju and D. R. Huang, "Low power consumption focusing actuator for a mini video camera," Journal of Applied Physics, vol. 99, no. 8, art. no. 08R901, 2006.

[10] S. K. Lin, C. M. Wang, and S. J. Wang, "Design and implementation of antihandshaking position control for a voice coil motor," Journal of Applied Physics, vol. 103, no. 7, art. no. 07F128, 2008.

[11] S. Southward, C. Radcliffe, and C. MacCluer, “Robust nonlinear stick-slip friction compensation,” ASME Journal of Dynamic Systems, Measurement and Control, vol. 113, no. 4,pp. 639–645, 1991.

[12] Jhih-Da Hsu, Ching-Lung Tsai and Ying-Yu Tzou, “Design and implementation of a voice-coil motor servo control IC for auto-focus mobile camera applications,” Power Electronics Specialists Conference 2007. PESC 2007.

[13] Taiwan R.O.C. Patent I287906.

[14] G. Ellis, Control System Design Guide, 2th Edition, ACADEMIC PRESS.

[15] Ned Mohan, Tore M. Undeland, William P. Robbins, Power Electronics, 3th Edition, John Wiley & Sons, Inc.

Appendix A : Control algorithm design flow

As shown in Figure 23, the first step of designing control algorithm is extracting the simulation model for the control plant. Ideally, we hope to get a complete mathematical model which describes all physical characteristics of the plant. But generally this ideal model is not available, only approximated model can be derived. So that experiments must be performed before claiming the control algorithm design is done, because control algorithm deals with the real control plant in experiment.

The developed control algorithm was firstly validated by simulation, the experiment then was performed. It can save a lot of time if validation experiment is performed on PC-based experimental environment before implementing it on FPGA. Following explains the improvement. Assume there are 3 kinds of control algorithms are evaluated and last one is the best choice. Also assume that it needs 1 day to implement algorithm in C and 10 days to implement algorithm on FPGA when m file algorithm is available. There will be 3 iterations before being ready for ASIC design. The time of performing algorithm validation is 1+1+1+10=13 days while adopting PC-based experimental environment. The time of performing algorithm validation without PC-based experimental environment is 10+10+10=30 days. It shows the proposed design flow can save 17 days in this case. If the target of experiment is not preparing for ASIC design, but just validating the algorithm with real control plant, which is mostly the case in school, then the tedious FPGA implementation thing can be ignored. We do think this is quit helpful for graduate students, because they are able to spend more time on algorithm studying.

We also recommend performing co-simulation of Simulink, ModelSim and PSIM, as shown in Figure 22, when implementing control algorithm with hardware description language (HDL). With this co-simulation environment, all functional bugs can be fixed easily because ModelSim can show all internal signals for the HDL design. On the contrary, debugging functional problem of HDL design in FPGA hardware is very difficult. Firstly, only few suspected signals can be routed to the IO pads because of the limitation of IO pads number. Not being able to show all suspected signals at once on a screen increases the

difficulty of debugging. It usually requires multiple iterations of pulling out suspected signals before a functional bug is fixed. Secondly it takes time to synthesis the HDL design to FPGA. It is really time consuming to debug functional problem of HDL design on the FPGA hardware by comparing doing this on the co-simulation environment.

Once FPGA validation is finished, the HDL design of the control algorithm is ready for ASIC implementation. And the people taking care of ASIC implementation can know nothing about algorithm things, but familiar with the ASIC design flow. Mostly, developing algorithm is the strength of the people staying in the educational fields. And ASIC implementation is the strength of the people staying in the industry fields. By adopting the proposed design flow, people in the school is able to release a "ready for ASIC design" HDL to a company for prototype and production. So that people in these two fields can cooperate seamlessly and be win-win partner.

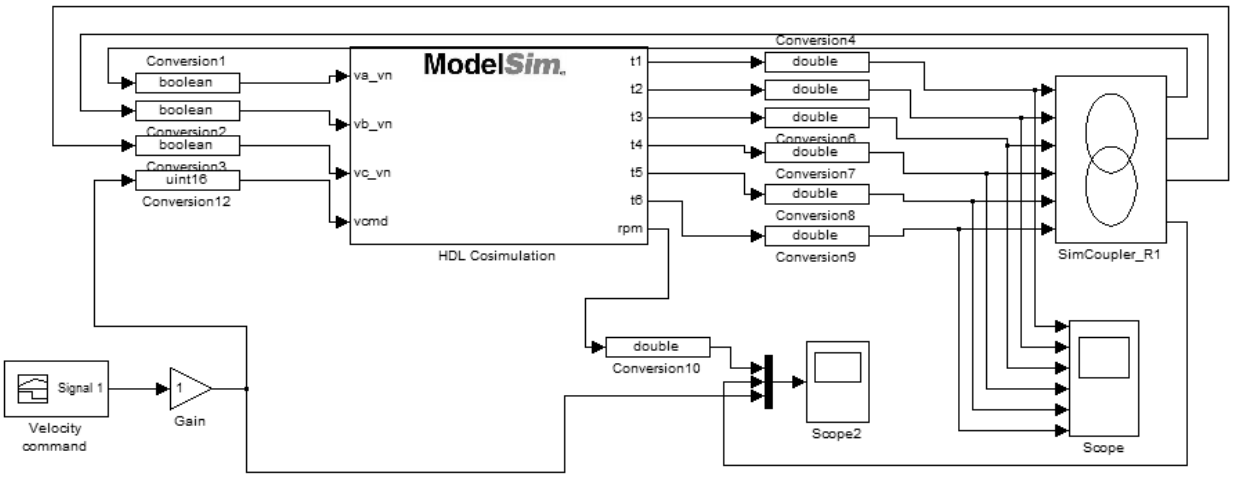


Figure 30 : Cosimulation environment for Simulink, Modelsim and PSIM.

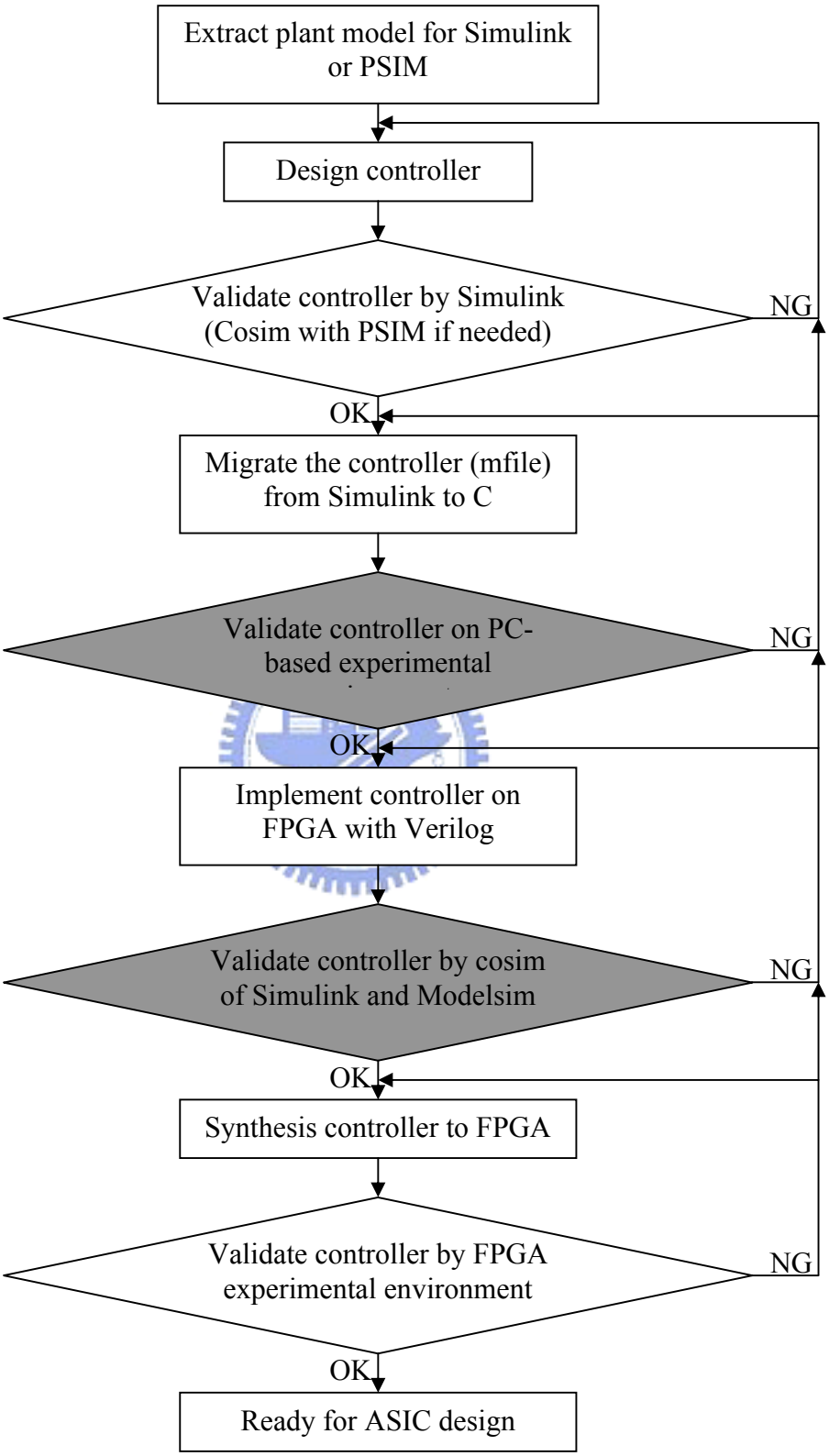


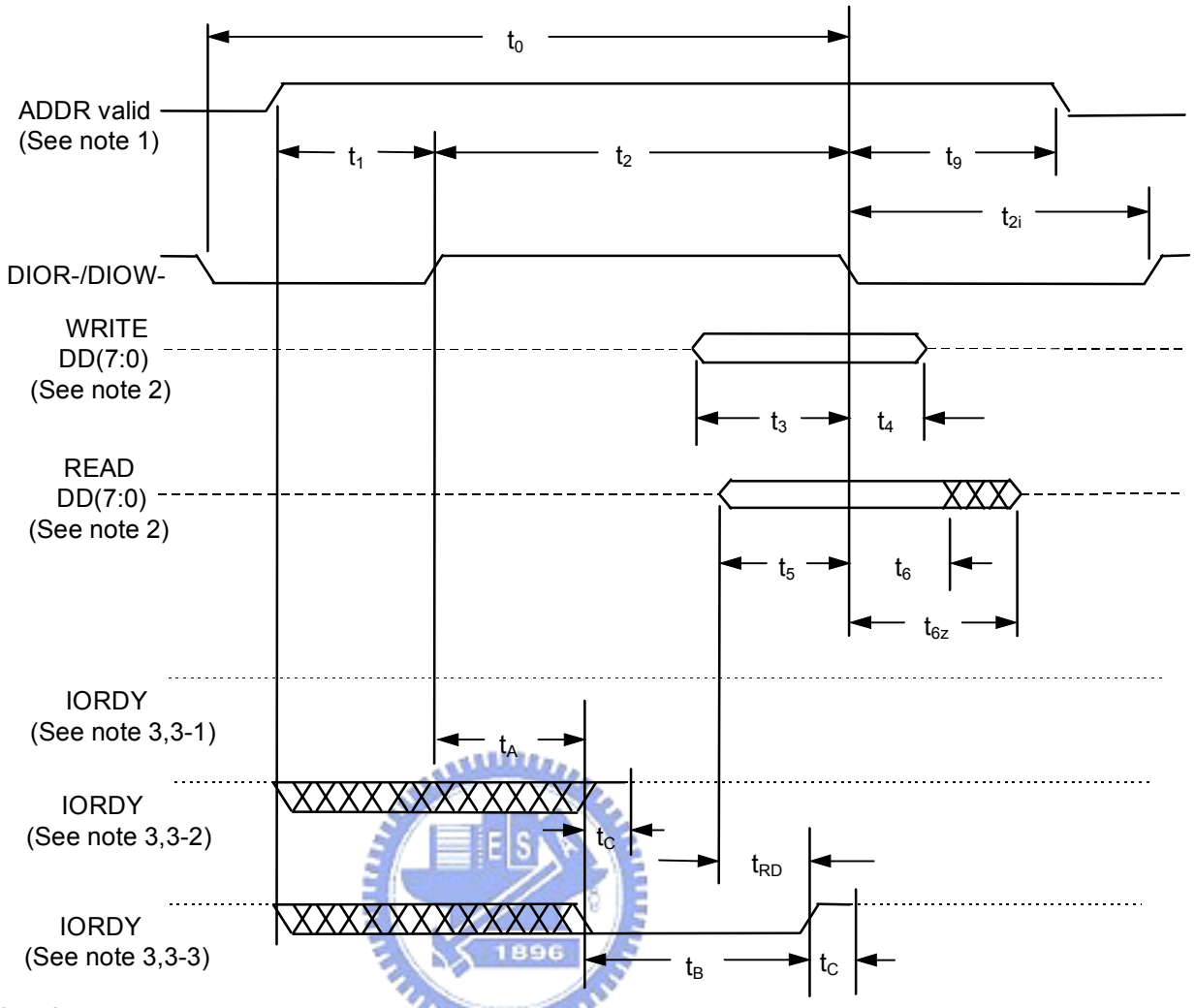
Figure 31 : The design flow of developing control algorithm.

Appendix B : PATA specification brief

The 40-pin ATA connector, with pin definition and signal description, and "Register transfer protocol" are shown here as a quit reference for implementation. Figure 25 shows the ATA connection orientation and pin number assignment. It can be seen that there are 2 keys for the orientation identification: 1) number 20 is an empty pin and 2) the housing has a gap. Table 2 shows the signals assignment for 40-pin ATA connector. The letter "-" of signal name indicates that the signal is low active. Figure 24 and Table 3 shows the timing of PIO register transfer protocol. It should be remarked that the high level means assertion in Figure 24. For example, DIOR- is low active. Assertion means that it is in low voltage. With this hardware regarding information, we can design a FPGA which is able to communicate with the PATA port on the motherboard.

Table 2 : Signals assignment for 40-pin ATA connector

Signal name	Connector contact		Signal name
RESET-	1	2	Ground
DD7	3	4	DD8
DD6	5	6	DD9
DD5	7	8	DD10
DD4	9	10	DD11
DD3	11	12	DD12
DD2	13	14	DD13
DD1	15	16	DD14
DD0	17	18	DD15
Ground	19	20	(keypin)
DMARQ	21	22	Ground
DIOW-	23	24	Ground
DIOR-	25	26	Ground
IORDY	27	28	CSEL
DMACK-	29	30	Ground
INTRQ	31	32	reserved
DA1	33	34	PDIAG-
DA0	35	36	DA2
CS0-	37	38	CS1-
DASP-	39	40	Ground



NOTES –

1 Device address consists of signals CS0-, CS1- and DA(2:0)

2 Data consists of DD(7:0).

3 The negation of IORDY by the device is used to extend the register transfer cycle. The determination of whether the cycle is to be extended is made by the host after t_A from the assertion of DIOR- or DIOW-. The assertion and negation of IORDY are described in the following three cases:

3-1 Device never negates IORDY, devices keeps IORDY released: no wait is generated.

3-2 Device negates IORDY before t_A, but causes IORDY to be asserted before t_A. IORDY is released prior to negation and may be asserted for no more than 5 ns before release: no wait generated.

3-3 Device negates IORDY before t_A. IORDY is released prior to negation and may be asserted for no more than 5 ns before release: wait generated. The cycle completes after IORDY is reasserted. For cycles where a wait is generated and DIOR- is asserted, the device shall place read data on DD(7:0) for t_{RD} before asserting IORDY.

Figure 32 : Timing of PIO register transfer protocol.

Table 3 : Timing value assignment for PIO register transfer protocol

	Register transfer timing parameters		Mode 0 ns	Mode 4 ns	Note
t_0	Cycle time	(min)	600	120	1,4
t_1	Address valid to DIOR-/DIOW- setup	(min)	70	25	
t_2	DIOR-/DIOW- pulse width 8-bit	(min)	290	70	1
t_{2i}	DIOR-/DIOW- recovery time	(min)	-	25	1
t_3	DIOW- data setup	(min)	60	20	
t_4	DIOW- data hold	(min)	30	10	
t_5	DIOR- data setup	(min)	50	20	
t_6	DIOR- data hold	(min)	5	5	
t_{6Z}	DIOR- data tristate	(max)	30	30	2
t_9	DIOR-/DIOW- to address valid hold	(min)	20	10	
t_{RD}	Read Data Valid to IORDY active (if IORDY initially low after t_A)	(min)	0	0	
t_A	IORDY Setup time		35	35	3
t_B	IORDY Pulse Width	(max)	1250	1250	
t_C	IORDY assertion to release	(max)	5	5	

NOTES –

1 t_0 is the minimum total cycle time, t_2 is the minimum DIOR-/DIOW- assertion time, and t_{2i} is the minimum DIOR-/DIOW- negation time. A host implementation shall lengthen t_2 and/or t_{2i} to ensure that t_0 is equal to or greater than the value reported in the devices IDENTIFY DEVICE data. A device implementation shall support any legal host implementation.

2 This parameter specifies the time from the negation edge of DIOR- to the time that the data bus is released by the device.

3 The delay from the activation of DIOR- or DIOW- until the state of IORDY is first sampled. If IORDY is inactive then the host shall wait until IORDY is active before the register transfer cycle is completed. If the device is not driving IORDY negated at the t_A after the activation of DIOR- or DIOW-, then t_5 shall be met and t_{RD} is not applicable. If the device is driving IORDY negated at the time t_A after the activation of DIOR- or DIOW-, then t_{RD} shall be met and t_5 is not applicable.

4 ATA/ATAPI standards prior to ATA/ATAPI-5 inadvertently specified an incorrect value for mode 2 time t_0 by utilizing the 16-bit PIO value

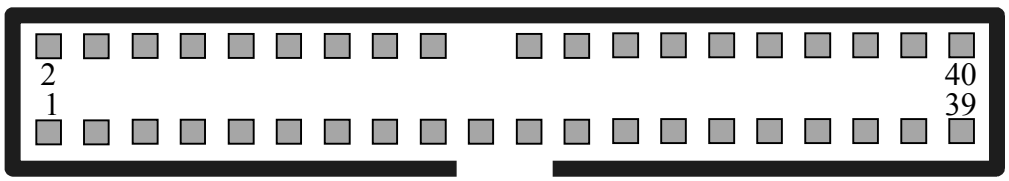


Figure 33 : ATA connector orientation and pin number assignment.

Appendix C : Verilog behavior coding of second order Butterworth filter

```
`timescale 1ns/1ns
//module butterworth;
 module butterworth_top(
  in,
  out
 );
  input [15:0]        in;
  output [15:0]       out;
  reg [15:0]          out;
  parameter           iir_a  = 7616, iir_b= 15232, iir_c= 7616, iir_d= -14929, iir_e= 6856;
  reg [31:0]          temp32,temp32_2, temp32_3, iir_o_n, iir_o_n_1, iir_o_n_2;
  reg [15:0]          iir_in_n_1, iir_in_n_2;
  wire [15:0]         iir_in_n = in;

  initial begin
        temp32 = 0; temp32_2 = 0; temp32_3 = 0; iir_o_n = 0; iir_o_n_1 = 0; iir_o_n_2 =
0;
        iir_in_n_1 = 0; iir_in_n_2 = 0;
        #10;

        Forever begin
              temp32 = $signed(iir_a) * $signed(iir_in_n);
              temp32 = $signed(iir_b) * $signed(iir_in_n_1) + $signed(temp32);
              temp32 = $signed(iir_a) * $signed(iir_in_n_2) + $signed(temp32);
              temp32_2=                    $signed(iir_d)                    *
$signed(downcase_f32_17_to_f16_17(iir_o_n_1));
              temp32_2=                    $signed(iir_e)                    *
$signed(downcase_f32_17_to_f16_17(iir_o_n_2)) + $signed(temp32_2);
              temp32_3=                    $signed(iir_d)                    *
$signed(downcase_f32_17_to_f16_2(iir_o_n_1));
              temp32_3=                    $signed(iir_e)                    *
$signed(downcase_f32_17_to_f16_2(iir_o_n_2)) + $signed(temp32_3);
               iir_o_n=case_f32_21_to_f32_17(temp32)                         -
case_f32_30_to_f32_17(temp32_2);
              iir_o_n= iir_o_n - case_f32_15_to_f32_17(temp32_3);
```

```
            out=downcase_f32_17_to_f16_0(iir_o_n);
            #25_000; //sampling time
            iir_in_n_2 = iir_in_n_1;
            iir_in_n_1 = iir_in_n;
            iir_o_n_2 = iir_o_n_1;
            iir_o_n_1 = iir_o_n;
        end //forever
end //initial

function [15:0] downcase_f32_17_to_f16_17;
        input [31:0]        i;
begin
        downcase_f32_17_to_f16_17 = {i[31],i[14:0]};
end
endfunction

function [15:0] downcase_f32_17_to_f16_2;
        input [31:0]        i;
begin
        if((~|i[31:30]) || (&i[31:30]) )
            downcase_f32_17_to_f16_2 = i[30:15];
        else begin
            if(i[31])
                downcase_f32_17_to_f16_2 = 16'h8000;
            else
                downcase_f32_17_to_f16_2 = 16'h7fff;
        end
end
endfunction

function [15:0] downcase_f32_17_to_f16_0;
        input [31:0]        i;
begin
        downcase_f32_17_to_f16_0 = {i[31],i[31:17]};
end
endfunction

function [31:0] case_f32_21_to_f32_17;
        input [31:0]        i;
```

```
  begin
        case_f32_21_to_f32_17 = {{4{i[31]}},i[31:4]};
  end
  endfunction

  function [31:0] case_f32_30_to_f32_17;
        input [31:0]        i;
  begin
        case_f32_30_to_f32_17 = {{13{i[31]}},i[31:13]};
  end
  endfunction

  function [31:0] case_f32_15_to_f32_17;
        input [31:0]        i;
  begin
        case_f32_15_to_f32_17 = {i[31],i[31:13]};
        if((~|i[31:29]) || (&i[31:29]) )
              case_f32_15_to_f32_17 = {i[31],i[28:0],2'b00};
        else begin
              if(i[31])
                    case_f32_15_to_f32_17 = 32'h8000_0000;
              else
                    case_f32_15_to_f32_17 = 32'h7fff_ffff;
        end
  end
  endfunction
endmodule
```

作者簡歷

個人資料

姓 名：蔡慶隆 (Ching-Lung Tsai)

生 日：民國 61 年 01 月 12 日

性 別：男

專 長：數位控制、順滑模態控制、IC 設計

學 歷

2007.9~2009.7 國立交通大學電機學院電機與控制學程碩士

1990.9~1994.6 國立中山大學電機系

1987.9~1990.6 高雄市立高雄高級中學