

PI Theory in Motor Control

This application note describes PI theory and PI in motor inverter control.

1 Introduction

1.1 Purpose

This application note describes PI in motor inverter control.

1.2 Definitions, Acronyms and Abbreviations

P Proportion

I Integral

1.3 Document Overview

The rest of document is organized as the following:

Chapter 2 explains the background of PI technical.

Chapter 3 explains the application in motor control.

Chapter 4 explains the application.

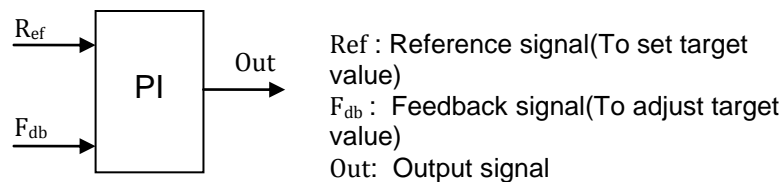
2 Technical Background

PI's theory

2.1 Overview

The PI loops use to faster and control motor.

Figure 1. Sample Module of PI Control

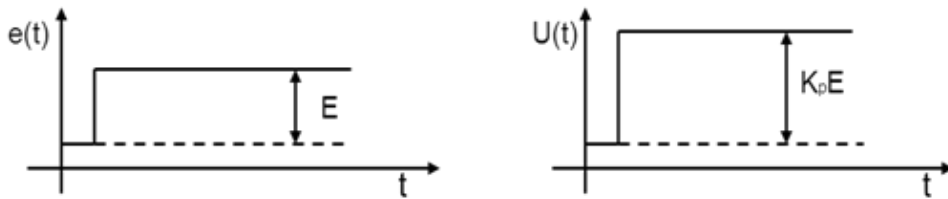


2.2 Theory

2.2.1 P- Proportion.

The Proportional term of the regulator is formed by multiplying the error signal by a P gain, using the PI regulator to produce a control response that is a function of the error magnitude. As the error signal becomes larger, the P term of the regulator becomes larger to provide more correction. In short, P is a magnify function.

Figure 2. P Regulator



In figure 2, we can know the P regulator has two characteristic:

1. Timely and faster control.
2. When the adjust process finish, but still have the error signal.

The P regulator formula is:

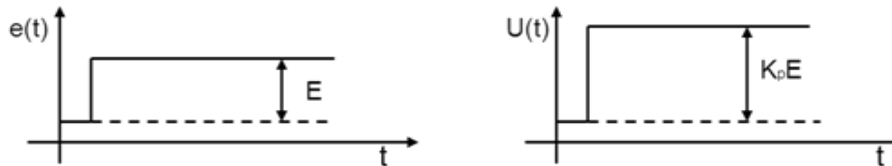
$$u(t) = K_p e(t) \quad (1)$$

Where $u(t)$ is output parameter, K_p is the proportion coefficient, $e(t)$ is the input parameter.

2.2.2 I-Integral

The Integral term of the regulator is used to eliminate small steady errors. The I term calculates a continuous running total of the error signal. Therefore, a small steady state error accumulates into a large error value over time. This accumulated error signal is multiplied by an I gain factor and becomes the I output term of the PI regulator.

Figure 3. I Regulator



In Figure 3, we can know the I regulator have five characteristic:

1. Not timely and faster control.
2. If have error signal, the regulator output control will increase, and it's speed always equal to start speed.
3. Control effect from strength to strength when time short and short.
4. I regulator can eliminate error signal.
5. In I regulator easy to bring adjust process surge.

The I regulator formula is:

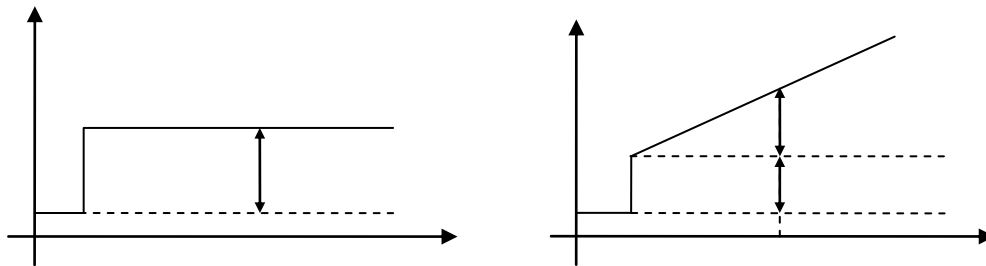
$$u(t) = k_I \int_0^t e(t) dt \quad (2)$$

Where $u(t)$ is output parameter, k_I is the integral coefficient, $e(t)$ is the input parameter, t is the integral time.

2.2.3 PI

The Proportional and Integral are PI. PI regulator is used to produce a control response that is a function of the error until disappear. As the error signal generate, regulator produce a control response of P, the I regulator calculates the error value over time. So, the PI regulator has P and I regulator's virtue, and eliminate themselves defect.

Figure 4. PI Regulator



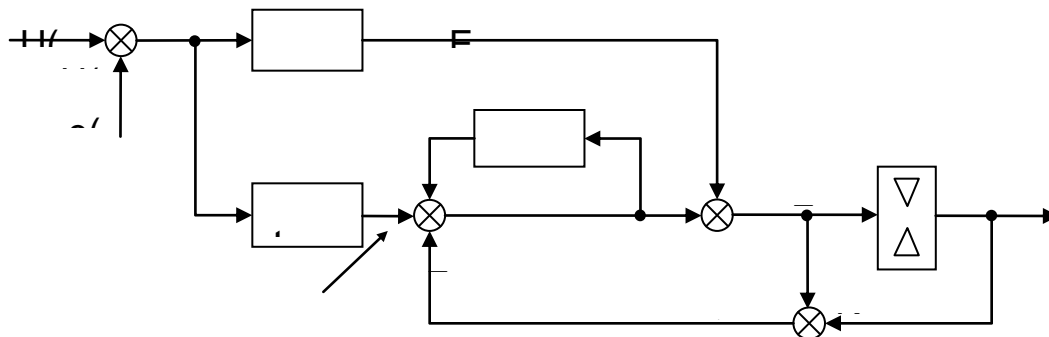
The I regulator formula is:

$$u(t) = u_p + u_i$$

$$u(t) = K_p \left(e(t) + \frac{1}{T} \int_0^t e(t) dt \right) \quad (3)$$

The PI regulator is a adjust module implemented with output saturation and with integral component correction. It can adjust input actual variable to follow the trace of target variable.

Figure 5. Implementation Process of PI Regulator as a Adjust Module



INPUT Y_{ref}, Y_{bf}

$$Y_e = Y_{ref} - Y_{fb}$$

$$U = X + K_p \cdot Y_e$$

$$\text{if}(U_{min} \leq U_k \leq U_{max})$$

$$U_k = U$$

$$\text{else if}(U_k > U_{max})$$

$$U_k = U_{max}$$

else

$$U_k = U_{min}$$

OUTPUT U_k

$$U_e = U - U_k$$

$$X = X \cdot Z^{-1} + K_i \cdot Y_e + U_e$$

3.2 Increment type PI loop in FM3 inverter platform

In FM3 inverter platform the PI loop type is increment type.

What is increment type PI loop? For example, define a array temp[10]={0,1,2,3,4,5,6,7,8,9 }, every time output the temp[k]= 0,1,2,3,4,5,6,7,8,9; Or output the temp[0], then output temp[1]=temp[0]+1, then output temp[2]=temp[1]+1; it's mean: this time output is last time output add the delta, this is the increment type PI loop.

$$u[k] = u[k - 1] + \{u[k] - u[k - 1]\}$$

Define:

$$\text{delta} = \{u[k] - u[k - 1]\}$$

Then:

$$u[k] = u[k - 1] + \text{delta}$$

Where: $u[k-1]$ is last time PI output , $u[k]$ is the PI output, delta is the PI error.

In math formula:

$$u_k = k_p \left(e_k + \frac{T}{T_i} \sum_{j=0}^k e_j \right) \quad (4)$$

$$u_{k-1} = k_p \left(e_{k-1} + \frac{T}{T_i} \sum_{j=0}^{k-1} e_j \right) \quad (5)$$

$$\Delta u = u_k - u_{k-1}:$$

$$\Delta u = u_k - u_{k-1} = k_p \left(e_k - e_{k-1} + \frac{T}{T_k} e_k \right) = k_p (e_k - e_{k-1}) + k_I e_k \quad (6)$$

Where: e_k is k time sample error value.

k_p is the proportion coefficient.

k_I is Integral coefficient, $k_I = k_p \frac{T}{T_k}$.

u_k is the output value.

Δu is the twice between border output error value.

4 Application

PI application achieve in system code

4.1 Function Description

```

/* active statistic task number */
Function Name:      PI_WeakenI

C file Name:        PI.C, PI.H
Function interface: Q15_VAL32 PI_WeakenI(volatile _stPIDPara *pPI_Q15)
  
```

```
typedef struct
{
    Q15_VAL32 Kp_Q15; // PI factors
    Q15_VAL32 Ki_Q15;
    Q15_VAL32 e_Q15; // PI input
    Q15_VAL32 e_Last1_Q15;
    Q15_VAL32 u_last1_Q15; // PI output
    Q15_VAL32 uMax_Q15;
    Q15_VAL32 uMin_Q15;
} _stPIDPara;
_stPIDPara *pPI.
```

Table 1. Input and Output of the Function

Item	Name	Description	Format
Inputs	e_Q15	This time error value input	Q15_VAL32
	e_Last1_Q15	Last error value input	Q15_VAL32
	Kp_Q15	P gain value	Q15_VAL32
	Ki_Q15	I gain value	Q15_VAL32
Outputs	u_last1_Q15	Last output	Q15_VAL32
	uMax_Q15	Output Integral limit maximal value	Q15_VAL32
	uMin_Q15	Output Integral limit minimal value	Q15_VAL32

4.2 Module usage

The following code is example for this module.

```
void example_PI_WeakenI ()
{
    Q15_VAL32 PI_output;
    pPI.e_Q15= INa;
    pPI.e_Last_Q15= INb;
    pPI.Kp_Q15=INc;
    pPI.Ki_Q15=INd;
    PI_output = PI_WeakenI (&pPI);
}
```

5 Document History

Document Title: AN205344 - PI Theory in Motor Control

Document Number: 002-05344

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	CCTA	03/24/2011	Initial Release
			04/07/2011	Redraw some picture and add explanation
			06/08/2012	Changed the format
*A	5045172	CCTA	01/05/2016	Migrated Spansion Application Note from MCU-AN-510105-E-12 to Cypress format
*B	5701669	AESATMP9	04/19/2017	Updated logo and copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6](#)

Cypress Developer Community

[Forums](#) | [WICED IOT Forums](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

©Cypress Semiconductor Corporation, 2011-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.