AN204394

FM0+ Family, 3-Phase ACIM Scalar Control with Slip Speed Control Solution

This application note introduces scalar control of a 3-phase ACIM solution with slip speed control scheme. The slip speed control scheme is known as an easy constructed, high efficiency, and medium dynamic response control algorithm.

Contents

1 Introduction

This application note introduces scalar control of a 3-phase ACIM solution with slip speed control scheme. The slip speed control scheme is known as an easy constructed, high efficiency, and medium dynamic response control algorithm.

In this document, the slip speed control is realized by capturing rotor speed by hall sensor to form speed close loop control. Further utilizing one channel ADC to sample DC bus voltage, SVPWM is generated. According to one PI regulator for voltage control, and one PI regulator for slip speed control, this scheme is realized for full speed range.

To prevent abnormal operations, over/under voltage protection, hall lost/lock rotor protection, and motor lose phase protection are dedicatedly designed and implemented.

1.1 Document Overview

The rest of this document is organized as below:

Chapter 2 explains System Hardware Environment

Chapter 3 explains Development Environment

Chapter 4 explains Firmware Introduction

Chapter 5 explains Getting Started

Chapter 6 explains Reference Documents

2 System Hardware Environment

Table 1. Demo System Hardware Environment

Name	Type	Description	Remark
MCU Series	S6E1A12B0AGP2	32-bit ARM Cortex-M0+ Core 32 Pins Maximum operating frequency: 40MHz On-chip flash memory: 88 Kbyte On-chip SRAM: 6 Kbyte	
IPM Module	SCM1559M	Maximum carrier frequency: 20 kHz Maximum main supply voltage: 400 V Maximum output current (continued): 10 A Logic supply voltage: 13.5~16.5 V Minimum dead time: 1.0 us	

3 Development Environment

Table 2. Development Environment

Name	Description	Part Number	Manufacturer	Remark
IAR Embedded Workbench 6.6	Code edit, compile, and online debug	N/A	N/A	N/A
Cypress Flash Loader	Flash download program	N/A	N/A	N/A
J-Link	IAR connection to target board	N/A	N/A	N/A
Eclipse	Code edit	N/A	N/A	N/A
Windows 7 Enterprise	Operation system (service pack 1, 64bit)	N/A	N/A	N/A

4 Firmware Introduction

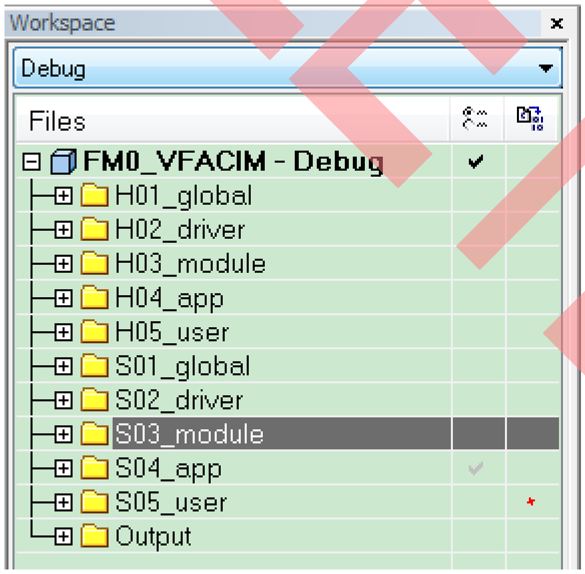
4.1 Firmware Features

Table 3. Firmware Features

No.	Feature	Description	Remark
1	Slip speed control	ACIM is controlled with optimized efficiency with slip speed controller.	Speed sensor is required.
2	Braking function	Braking down motor with no additional hardware.	
3	Field weakening control	Speed range of motor is extended by slip speed regulation.	
4	Bi-direction rotation	Both clockwise and counter-clockwise rotations are OK.	
5	Over current protection	Hardware over current protection. (Software protection is available if current sensor is implemented)	
6	Voltage protection	Irregular HIGH/LOW voltage protection, normal over/under voltage protection.	
7	Hall lost/Lock rotor protection	Prevent long time operation in no speed signal case.	
8	Phase lose protection	Motor phase lose case will be protected.	
9	Variable carrier frequency	Maximum 16kHz	
10	UART communication	UART communication with host machine	
11	OOB function	Out-of balance for washing machine application	
12	Weighting function	Weight measurement for washing machine application	

4.2 Firmware Architecture

Figure 1. Overview of Firmware File System



4.2.1 Firmware Execution Flow

Figure 2 shows the execution flow of firmware that controls an ACIM by slip speed controller.

In main function, after MCU and control system are initialized, a while loop is executed that starts/stops motor according to user command speed. In another hand, UART communication and timer event are processed for user defined operations.

In normal control occasion, three main interrupt routines are executed. FRT zero interrupt occurs at a frequency named carries frequency, and once FRT interrupt happens, an ADC interrupt is triggered to sample DC bus voltage and current. After AD sample, motor control is executed in FRT interrupt routine, with the knowledge of sampled voltage and rotor speed. Another interrupt is triggered by Hall signal to capture rotor speed, and both the falling edge and rising edge are captured to calculate rotor speed.

In abnormal ISR routines, a DTTI interrupt is triggered by a fault signal from IPM, and activates hardware over current protection and immediately stop the motor. The WDT(watch dog timer) interrupt is triggered by both hardware and software WDT, and once this interrupt is triggered, motor will be stopped immediately.

Figure 3 shows the block diagram of slip speed control system. For more information of slip speed control, please refer to S6E1A1_AN710-00001-1v0-E-3Phase_ACIM_Scalar_Control.pdf.

Figure 2. Firmware Execution Flow

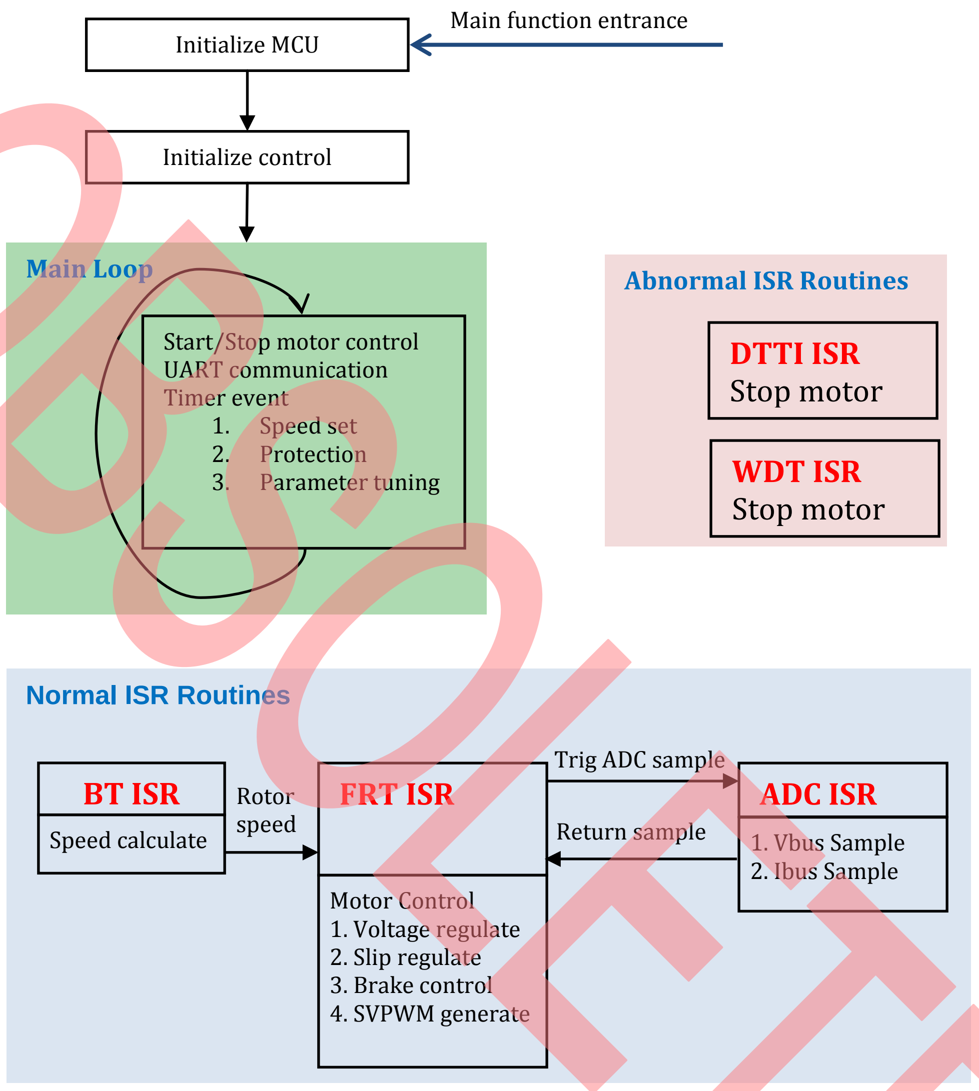
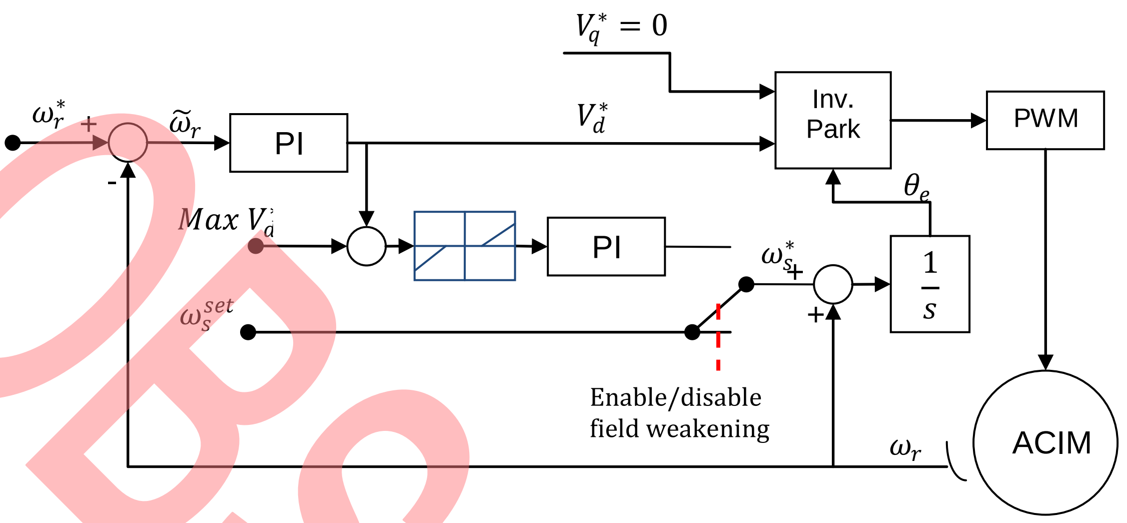


Figure 3. Block Diagram of Slip Speed Control System



4.2.2 Files Description

Figure 4. Firmware Layers Introduction

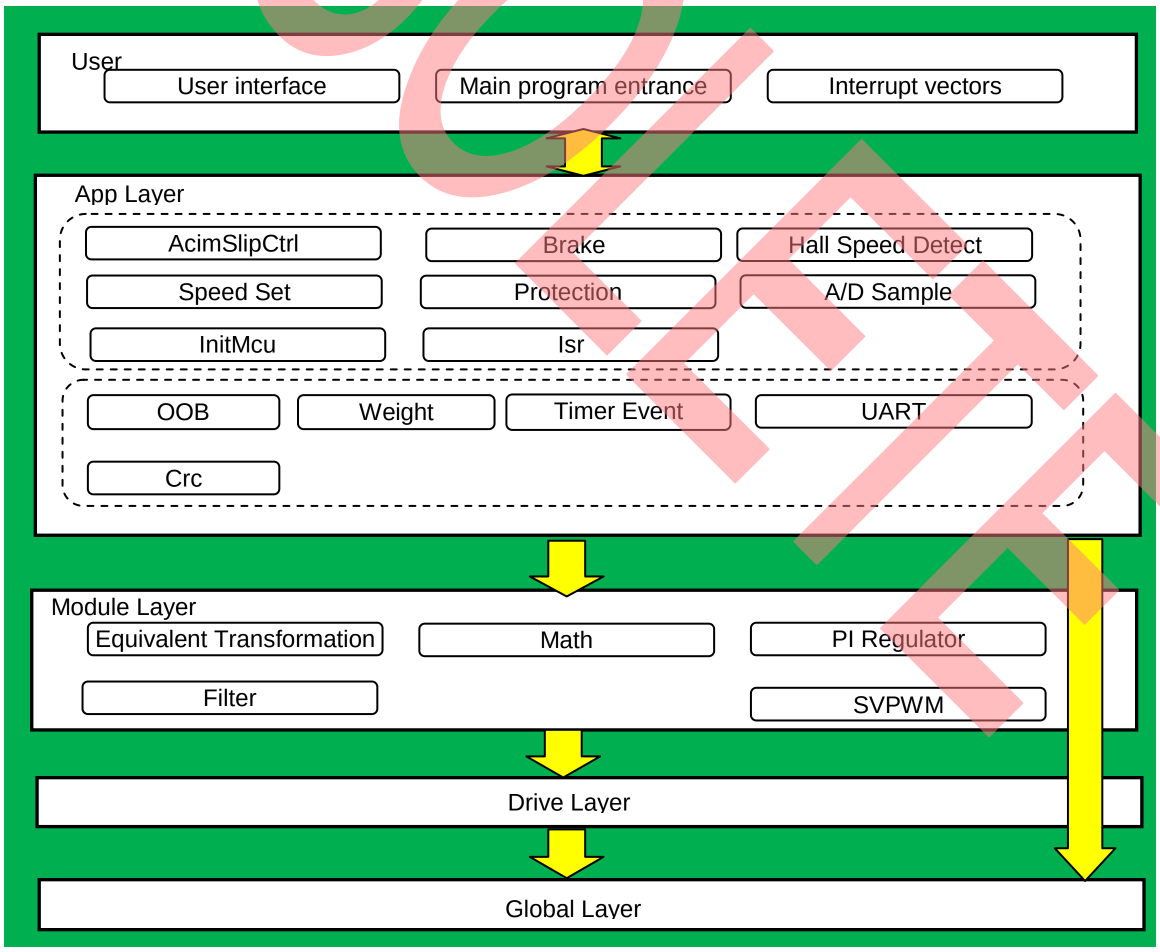


Table 4. Source Files Description

Layer	Files	Description
S01_global	interrupts.c	MCU related files.
	pdl.c	
	system_s6e1a1.c	
S02_driver	adc.c	MCU drivers.
	bt.c	
	hwwdg.c	
	mft_adcmp.c	
	mft_frt.c	
	mft_ocu.c	
	mft_wfg.c	
	swwdg.c	
S03_module	EquivalentTransformation.c	Basic functions for motor control.
	Filter.c	
	Math.c	
	Pid.c	
	PidReg_cm0_64bit.asm	
	Scpwm_cm0.asm	
S04_app	AdcSample.c	ADC sample parameter initialization and offset check.
	Brake.c	Braking function of ACIM.
	crc.c	crc check for UART communication
	HallSpeedDetect.c	Hall speed detection called in BT ISR for speed measurement.
	InitMcu.c	Initialize MCU peripherals.
	Isr.c	ISR interrupt entrance.
	MotorCtrl.c	Motor control files, including initialization, startup motor, stop motor, and FRT routine called by FRT ISR.
	OOB.c	OOB function of washing machine.
	Protection.c	Protection functions, including voltage protection, Hall lost/Lock rotor protection, and lose phase protection
	SpeedSet.c	Command and target speed set functions.
	TimerEvent.c	Timer triggered functions, including speed set, protection, and parameter self-adjustment.
	UART.c	UART communication
	Weight.c	Weight measure of washing machine
S05_user	Main.c	main function
	startup_s6xxxx.s	Vector table of target MCU.
	CustomerInterface.c	Customer interface for parameter configuration.

The firmware consists with five layers, and each layer places header files and source files separately. Figure 4 shows the functions of each layer, and Table 4 describes the details.

5 Getting Started

5.1 Hardware Configuration

Figure 5. Top View of Demo PCB

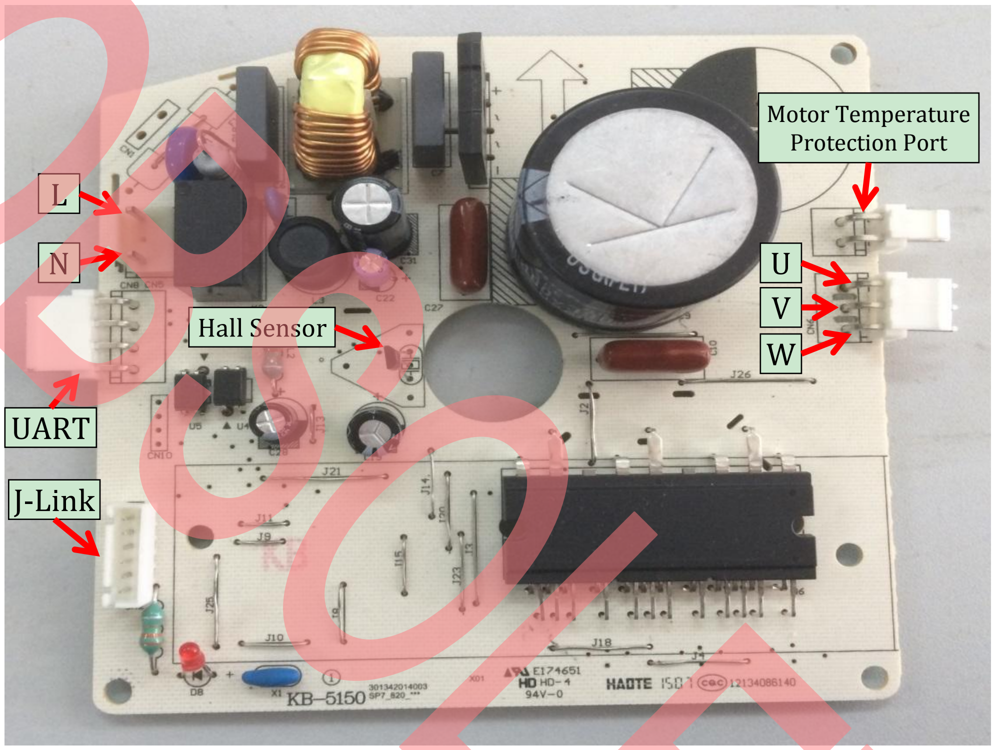


Figure 5 shows the top view of demo PCB. First connect motor cables and motor temperature protection port. If there is no temperature port, short this port. Insert J-Link wire, and connect AC cables. Now the PCB is ready for your debug.

If your hardware is different from demo PCB, please configure your hardware infomation in H05_user\HardwareConfig.c.For further information of your hardware, please find H04_user\InitMcu.cto adapt your hardware.

Figure 6. Hardware Information Configuration

```
#define ADC_VOLT_REF         5.0f          //reference voltage of ADC sample
#define ADC_VALUE_MAX        4096.0f       //4096 for 12bit ADC
#define VDC_FACTOR           93.3          //gain of DC bus voltage samplling
#define ADC_CH_VDC           2             //ADC channel of DC bus voltage

#define ADC_CH_I_BUS         1             //ADC channel of DC bus current
#define ADC_I_BUS_FACTOR     15.6          // gain of DC bus current sampling
#define ADC_I_BUS_BIAS       2048          // bias of DC bus current sampling
// configure hall I/O and Pin
#define USER_BT_TIMER_SIZE   PwcSize16Bit; //base timer PWC size
#define HALL_A_TIMER_LOW_CH  FM0P_BT0_PWC  //base timer unit for hall speed
/** UART Select */
#define  FM0P_MFSx_UART            FM0P_MFS3_UART //MFS unit for UART communication
```

5.2 Firmware Configuration

Before startup your motor, please configure your control target to make sure motor is operated correctly. You can find firmware configuration parameters in *S05_user\CustomerInterface.c.*

5.2.1 Basic Drive Parameters

```
//
// carrier frequency and dead time
//
uint16_t    MotorCtrl_u16CarrierFreq    = 16000;    // carrier frequency (Hz)
float32_t   MotorCtrl_f32DeadTimeUs     = 2.0f;     // Dead timer us
uint32_t    RelayDelayOnTms              = 2000;     // the time delay for relay
switched
                                                    //on,unit:ms
```

First of all, select your carrier frequency and dead time for motor drive. Set a time delay for relay open when system power is on.

5.2.2 Motor Parameters

```
//
// motor parameters
//
int32_t     MotorCtrl_i32PolePairs  = 2;        // pole pairs
float32_t   Motor_f32TransRate      = 11.6;     // TransRate of motor
float32_t   MotorCtrl_f32MaxVf      = 3.0;      // maximum V/f ratio
int32_t     MotorCtrl_i32RpmMin     = 30;       // minimum speed
int32_t     MotorCtrl_i32RpmMax     = 1200;     // maximum speed
```

5.2.3 Hall Sensor Parameters

```
//
// hall speed sensor
//
uint32_t    MotorCtrl_u32EdgesPerCycle     = 16;// hall edges per motor cycle
float32_t   MotorCtrl_f32BaseTimerClkMHz   = 10.0;// Base timer clock
```

In this firmware, motor speed is measured from a hall sensor, and captured by base timer. The base timer captures both the falling edge and rising edge of hall signal. For example:

Your hall sensor returns 8 pulses per mechanical cycle, then you have 8 falling edges and 8 rising edges, therefore 16 edges per cycle is set.

5.2.4 Startup Parameters

```
//
// motor start-up settings
//
int32_t     MotorCtrl_i32StartRpm1          = 100;
int32_t     MotorCtrl_i32StartRpm2          = 400;
float32_t   MotorCtrl_f32StartWsHz          = 8.0;
int32_t     MotorCtrl_i32StartAccRpmPerSec  = 1500;
```

The startup process applies slip speed curve for large startup torque. This section defines the transition point at startup as well as speed acceleration rate.

5.2.5 Normal Running Parameters

```
//
// motor normal running settings
//
float32_t   MotorCtrl_f32MinWsHz            = 2.8;  // (Hz) nominal slip speed
float32_t   MotorCtrl_f32MaxWsHz            = 18.0; // (Hz) maximum slip speed
```

The normal running operation requires two slip speeds. One is for low speed and medium speed region that applies MTPA control, named as minimum slip speed. Another is for field weakening control, which defines the maximum slip speed allowed in field weakening region.

5.2.6 Braking Parameters

```
//
// motor  brake settings
//
float32_t   MotorCtrl_f32FreeBrakeVsRamp    = 25.0;     // Volt/second
float32_t   MotorCtrl_f32ForceBrakeVs       = 19.0;     // Volt
float32_t   MotorCtrl_f32ForceBrakeVsRamp   = 200.0;    // Volt/second
```

The braking process has two steps. First when motor is under AC excitation, stator voltage will decrease to zero with voltage ramp defined by `MotorCtrl_f32FreeBrakeVsRamp`. If force brake is enabled in braking process, a DC voltage with amplitude `MotorCtrl_f32ForceBrakeVs` will be imposed on stator, and the rising ramp of this voltage is defined by `MotorCtrl_f32ForceBrakeVsRamp`.

5.2.7 Field Weakening Parameters

```
//
// field weakening control
//
float32_tMotorCtrl_f32FwUpperVsK     = 0.97;
float32_t   MotorCtrl_f32FwLowerVsK  = 0.91;
```

Two voltage ratio of maximum output voltage is defined for field weakening control, 0.90~0.99 are recommended for these two values.

5.2.8 PI Parameters

```
//
// PI parameters
//
float32_t   MotorCtrl_f32SpdRegTimeMs   = 0.6;  // speed regulation cycle (ms)
// voltage PI regulator
int32_t     MotorCtrl_i32LowSpdRpm      = 1600;// low speed transition speed
int32_t     MotorCtrl_i32HighSpdRpm     = 3000;// high speed transition speed
float32_t   MotorCtrl_f32LowSpdVsKp     = 2.5;// low speed Kp of voltage PI
float32_t   MotorCtrl_f32LowSpdVsKi     = 6;// low speed Ki of voltage PI

float32_t   MotorCtrl_f32HighSpdVsKp    = 0.5;// high speed Kp of voltage PI
float32_t   MotorCtrl_f32HighSpdVsKi    = 1.0;// high speed Kpof voltage PI

// slip speed PI regulator
float32_t   MotorCtrl_f32WsKp           = 0;// Kp of slip speed PI for field weaken
float32_t   MotorCtrl_f32WsKi           = 1;// Ki of slip speed PI for field weaken
```

5.2.9 Protection Parameters

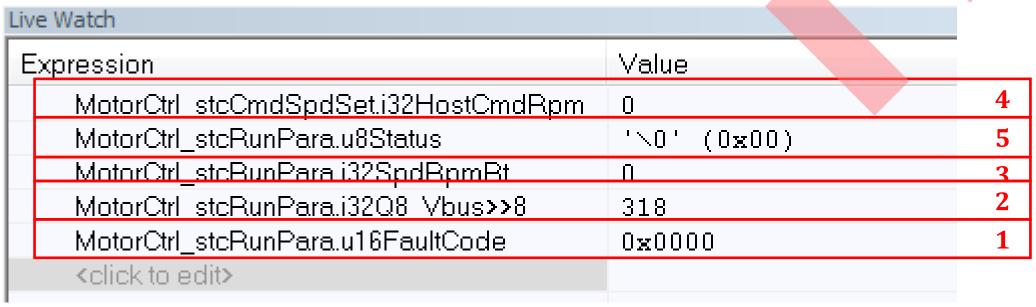
```
//
// protections
//

int32_t     MotorCtrl_i32OverVoltage    = 360;      // over voltage threshold
int32_t     MotorCtrl_i32UnderVoltage   = 200;      // under voltage threshold

int32_t     MotorCtrl_i32IrreHighVoltage = 390;// irregular high voltage limit
int32_t     MotorCtrl_i32IrreLowVoltage  = 100;    // irregular low voltage limit
```

In this firmware, over current protection is done by hardware, thus you only need to configure voltage protection in customer interface. There are two limits for both over voltage and under voltage protection. The irregular high voltage and irregular low voltage take a shorter time to trigger a protection than normal over voltage and under voltage.

5.3 Debug Slip Speed Control System

5.3.1 Check Your Hardware

Figure 7. Live Watch of IAR System



Live Watch

Expression	Value	
MotorCtrl_stcCmdSpdSet.i32HostCmdRpm	0	4
MotorCtrl_stcRunPara.u8Status	'\0' (0x00)	5
MotorCtrl_stcRunPara.i32SpdRpmRt	0	3
MotorCtrl_stcRunPara.i32Q8_Vbus>>8	318	2
MotorCtrl_stcRunPara.u16FaultCode	0x0000	1
<click to edit>		

After first time burning the firmware into hardware, check below items to make sure both your hardware and firmware are correctly configured:

1. Check item 1 in Figure 7. If fault code is 0x0000, no error code is detected. Otherwise refer to Figure 8 to fix the problem.
2. Check item 2 in Figure 7. This item is DC bus voltage. If this value is in your expectation, then DC bus sample is OK. Otherwise check your ADC configuration and sample parameters.
3. Check item 3 in Figure 7.This item is rotor speed measured by Base Timer. In standstill case, this value should be zero. Rotate your motor, and keep watching this value, it becomes a non-zero integer when motor is rotating. If this value keeps being zero, check your Hall sensor and Base Timer configuration.

5.3.2 Startup Your Motor

Once the hardware and firmware are properly configured, you are ready to rotate your motor.

Input a speed into item 4 in Figure 7, this is command speed. At first time, run the motor in low speed region, and check whether the parameters are proper for your motor.

You might confront below problems:

1. Fault Code: 0x0008

This means hardware over current protection is triggered. This is probably due to that a large V/f ratio caused over current protection; you can smaller the maximum V/f ratio as 5.2.2 described. Otherwise, check your hardware that whether you have set a too low value to hardware over current protection.

2. Fault Code: 0x0400

This is lock rotor protection or hall lost protection. Double confirm motor can be freely rotates, and hall sensor are correctly configured. Otherwise, refer to next problem.

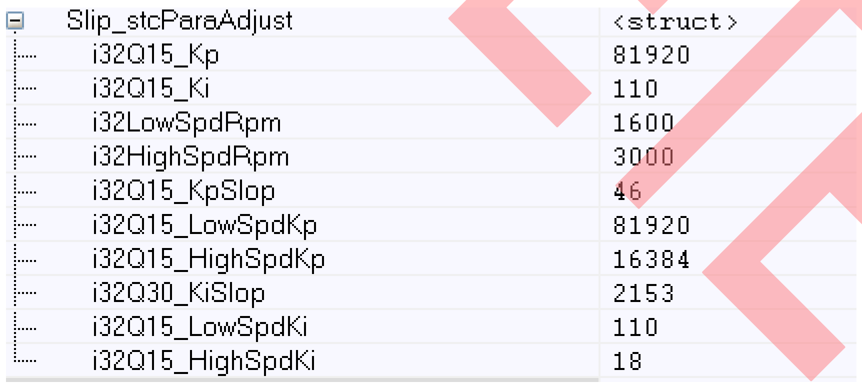
3. Motor does not rotate with non-zero command speed

If current appears but motor never rotates. This is probably that the maximum V/f ratio is too small to startup the motor; you can enlarge the maximum V/f ratio as 5.2.2 described.

5.3.3 Tuning PI Parameters

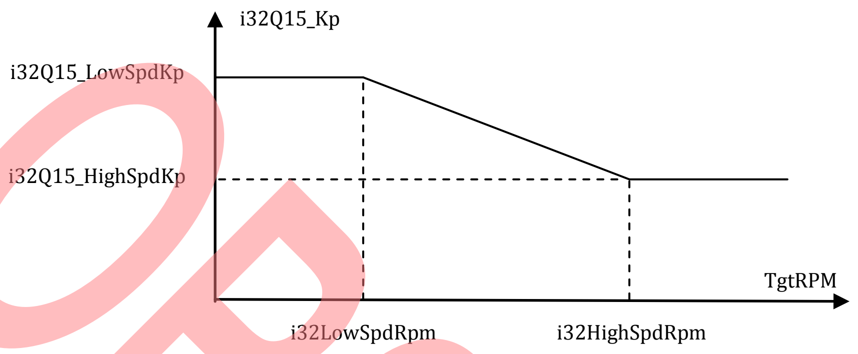
Find the PI parameter for voltage regulation as Figure 8 shows. Tune the PI parameters such that both the dynamics and stability satisfies your requirement. Figure 9 shows the auto tuning process of Kp according to target speed, and Ki shares the same adjustment pattern.

Figure 8. Voltage PI Regulation Parameters



Slip_stcParaAdjust	<struct>
i32Q15_Kp	81920
i32Q15_Ki	110
i32LowSpdRpm	1600
i32HighSpdRpm	3000
i32Q15_KpSlop	46
i32Q15_LowSpdKp	81920
i32Q15_HighSpdKp	16384
i32Q30_KiSlop	2153
i32Q15_LowSpdKi	110
i32Q15_HighSpdKi	18

Figure 9. Auto-Tuning of Kp According to Target Speed

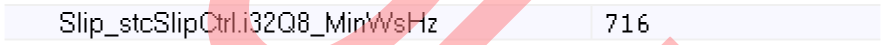


5.3.4 Tuning Slip Speed

The slip speed is the key factor to optimized power consumption operation. The theoretical optimal slip speed is calculated as

$$\omega_s = \frac{R_r}{L_r}$$

This means that the optimal slip speed equals to the inverse of rotor time constant. Now drive the motor in a proper speed stably (no disturbance is preferred). Be aware that speed should not be too slow (or efficiency is quite low) or too high (otherwise field weakening is introduced). Find the minimum slip speed in live watch window



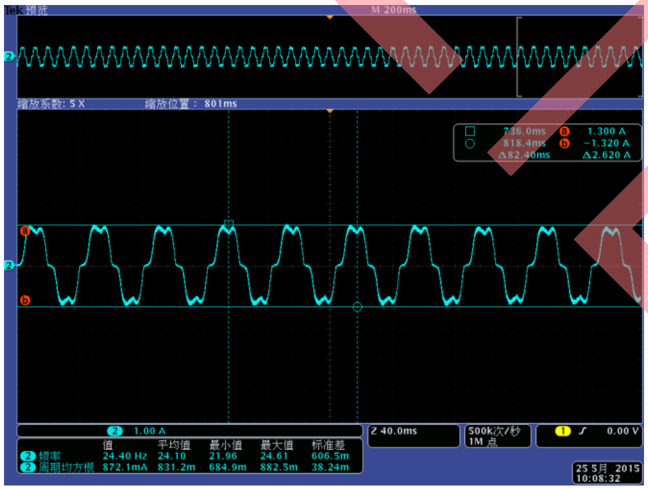
```
Slip_stcSlipCtrl.i32Q8_MinWsHz    716
```

Tune the parameter and observe phase current or input power. The optimized slips speed is found when minimum current or power is achieved. Record this slip speed and use this slip speed for your further application.

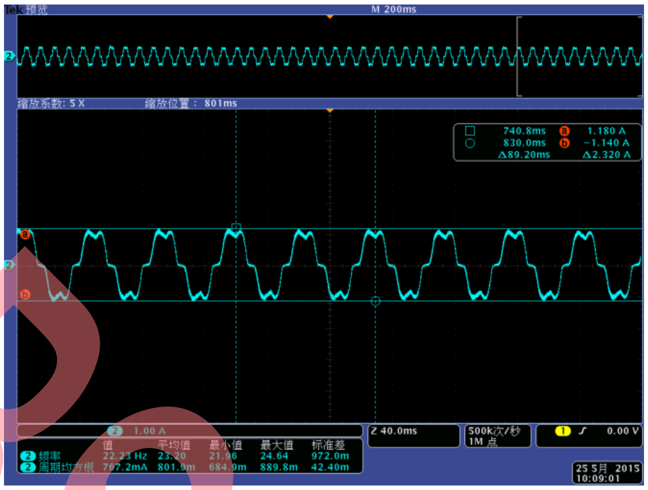
Figure 10 shows the current waveform of motor at 580 RPM with different slip speed. Apparently, the target motor runs under slip speed equals to 2.8Hz approached a better efficiency.

Figure 10. 580 RPM with Different Slip Speed

a. $\omega_s = 5.0$ Hz



b. $\omega_s = 2.8$ Hz



5.3.5 Tuning High Speed Region

The high speed region involves field weakening control. Due to the lack of current close-loop control, you are going to tuning below parameters to achieve a proper performance. Table 5 shows each parameter that relates to field weakening control.

Table 5. High Speed Region Tuning Parameters

Parameter	Customer Interface Variable	Description
Slip_stcWsPid.i32Q15_Kp	MotorCtrl_f32WsKp	Kp of slip speed PI regulator
Slip_stcWsPid.i32Q15_Ki	MotorCtrl_f32WsKi	Ki of slip speed PI regulator
Slip_stcSlipCtrl.i32Q8_LowerVsK	MotorCtrl_f32FwLowerVsK	Lower voltage threshold for field weakening control. Tuning target: 1. Field weakening is fluent and stably exited.
Slip_stcSlipCtrl.i32Q8_UpperVsK	MotorCtrl_f32FwUpperVsK	Upper voltage threshold for field weakening control. Tuning target: 1. Field weakening enters fluently 2. Field weakening is effectively extending speed range.
Slip_stcSlipCtrl.i32Q8_MaxWsHz	MotorCtrl_f32MaxWsHz	Maximum allowed slip speed for field weakening control. Tuning target:: 1. Speed range is extended 2. Current range is within rated value with slip speed equals to this value.

5.3.6 Tuning the Braking Function

Figure 11 shows the command voltage in braking process. Three parameters are to be tuned for this function.

1. MotorCtrl_f32FreeBrakeVsRamp

This is a voltage ramp in unit V/s that decreases the amplitude of AC voltage. Tune this ramp such that the DC bus voltage does not boost too much and the decreasing process as short as possible.

2. MotorCtrl_f32ForceBrakeVsRamp

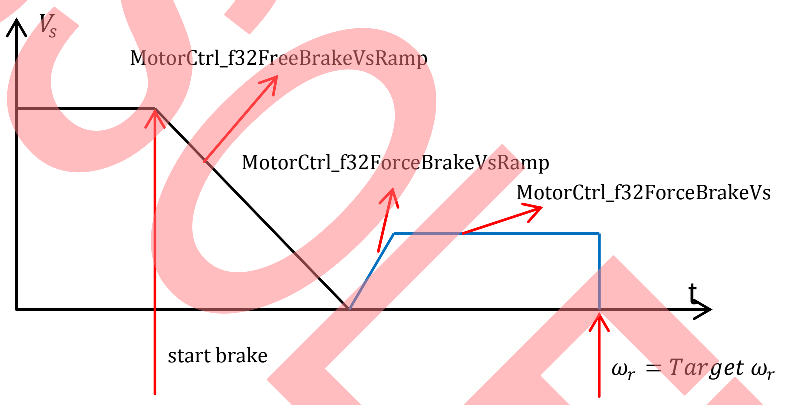
This is a voltage ramp in unit V/s that increases a DC voltage imposed to stator. Choose a proper one that current reaches to its stable point quickly with small overshoot.

3. MotorCtrl_f32ForceBrakeVs

This is the braking DC voltage imposed on stator. Choose a proper voltage that rotor brakes fast and current amplitude is within the rated range is OK.

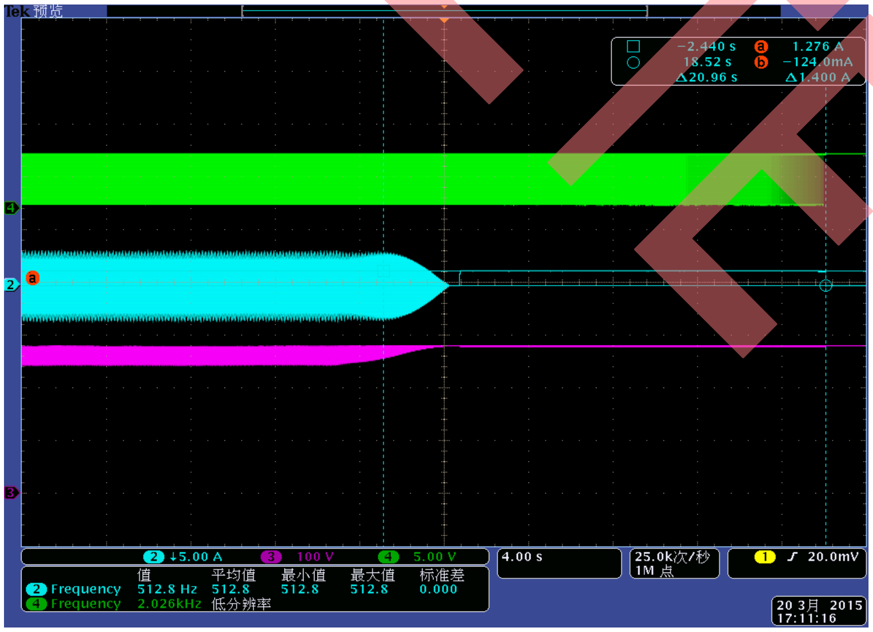
Figure 12 shows a typical braking process with force braking. Channel 2 is stator current, channel 3 is DC bus voltage, and channel 4 is hall signal.

Figure 11. Command Voltage in Braking Process



V_s
MotorCtrl_f32FreeBrakeVsRamp
MotorCtrl_f32ForceBrakeVsRamp
MotorCtrl_f32ForceBrakeVs
t
start brake
$\omega_r = Target\ \omega_r$

Figure 12. A Typical Braking Process



```
□   −2.440 s   a    1.276 A
○    18.52 s   b   −124.0mA
    Δ20.96 s        Δ1.400 A
```

```
2 ↓5.00 A    3 100 V    4 5.00 V    4.00 s    25.0k次/秒    1 ⨍ 20.0mV
                                              1M 点
              值          平均值   最小值   最大值   标准差
2 Frequency   512.8 Hz    512.8    512.8    512.8    0.000
4 Frequency   2.026kHz    低分辨率
                                                       20 3月 2015
                                                       17:11:16
```

5.4 Troubleshooting

For other unstated fault case, please refer to fault code in Figure 13 to fix problems.

Figure 13. Fault Code of Slip Speed Control System

```
#define NORMAL_RUNNING       0x0000  //no error
#define AD_MIDDLE_ERROR      0x0001  //current sample 2.5V offset error
#define SW_OVER_CURRENT      0x0002  //over-current of FW
#define SINK_ERR             0x0004  //IPM circuit fault
#define MOTOR_OVER_CURRENT   0x0008  //over-current of HW
#define OVER_VOLTAGE         0x0010  //DC bus over-voltage
#define UNDER_VOLTAGE        0x0020  //DC bus under-voltage
#define POWER_OVER           0x0040  //motor over-power
#define IPM_TEMPOVER         0x0080  //IPM temperature error
#define MOTOR_TEMPOVER       0x0100  // motor over temperature
#define MOTOR_LOSE_PHASE     0x0200  //motor lose phase
#define MOTOR_LOCK           0x0400  //motor lock
#define DCBUS_ERR            0x0800  // dc bus voltage error
#define COMM_ERROR           0x1000  //communicate error code
#define SF_WTD_RESET         0x2000  //FW watch dog reset
#define HW_WTD_RESET         0x4000  //HW watch dog reset
#define UNDEFINED_INT        0x8000  //undefined interrupt
```

6 Reference Documents

1. FM0+ S6E1A1 Series Data Sheet, Revision 0.1, 2013
2. FM0+ Family Peripheral Manual, Revision 1.0, 2014
3. FM0+ Family Timer Part Peripheral Manual, Revision 1.0, 2014
4. FM0+ Family Analog Macro Part Peripheral Manual, Revision 1.0, 2014
5. P. C. Krause, O. Wasynczuk, and S. D. Sudhoff, Electric Machinery and Drive Systems. IEEE Press, 2002

7 Additional Information

For more Information on Cypress semiconductor products, visit the following websites:

http://www.cypress.com/cypress-microcontrollers

8 Document History

Document Title: AN204394 - FM0+ Family, 3-Phase ACIM Scalar Control with Slip Speed Control Solution

Document Number: 002-04394

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	CBZH	06/15/2015	Initial release
*A	5100364	CBZH	06/16/2016	Migrated Spansion Application Note AN710-00009-1v0-E to Cypress format. There is no web link for the reader to get the PCB (evaluation board), So this AN is for obsolete.

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
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. 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

Phone	: 408-943-2600
Fax	: 408-943-4730
Website	: www.cypress.com