Application Report
SPRAC84–May 2017

Fan Inverter Using PMSM Sensorless FOC Algorithm, Compliant With IEC 60730 (UL1998), Based on TMS320F28021 MCU

Terry Deng

ABSTRACT

This application note presents a detailed solution to implement in fan motor applications based on TMS320F28021 microcontrollers. The solution uses the PMSM sensorless FOC algorithm, and also designed to comply with the IEC 60730 (UL1998) standard.

TMS320F2802x devices are part of the family of C2000 microcontrollers, which enable cost-effective design of intelligent controllers for 3-phase motors by reducing the system components, and also increase efficiency. With these devices it is possible to realize far more precise digital vector control algorithms like the field oriented control (FOC). Implementation of this algorithm is described in this document. The FOC algorithm maintains efficiency in a wide range of speeds, and takes into consideration torque changes with transient phases by processing a dynamic model of the motor. Among the proposed solutions are ways to eliminate the phase current sensors and use an observer for speed sensorless control.

Manufacturers of household appliances must take steps to ensure safe and reliable operation of their products to meet the IEC 60730 standard. The IEC 60730 standard covers mechanical, electrical, electronic, EMC, and abnormal operation of AC appliances. Most home appliances including washing machines, dryers, refrigerators, freezers, and cookers or stoves fall into the Class B classification. How to design solutions to fulfill Class B compliance is discussed in this document. Also, the document lists which MCU components must be tested, which faults must be detected, and the appropriate reactive measures.

Copyright © 2017, Texas Instruments Incorporated

Contents

1 Introduction 3
2 System Specification 3
3 MCU Use Overview 4
4 Hardware Design 5
5 Firmware Design 8
6 Fault Handle 11
7 Microelectronic Fault Protection Function 15
8 Final Product Performance 18
9 Related Documentation 20

List of Figures

1 System Overall Structure 5
2 Three-Phase Driver Circuit 6
3 Current Sample Circuit 7
4 Comparator Protection Circuit 7
5 FOC Algorithm Structure 8
6 Main Procedure Flow 9
7 Overcurrent Protection Flow 11
8 Loss Phase Protection Flow 12
9 Lock of Rotor Protection Flow 13
10 System Config Header File 15
11 Error Simulation Header File 15
12 Power Consumption Record 18
13 60-Hz Current Waveform 19
14 100-Hz Current Waveform 19

List of Tables

1 System Specifications 3
2 MCU Pin Assignment 4

Trademarks

Piccolo is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

Copyright © 2017, Texas Instruments Incorporated

1 Introduction

This application report presents a fan inverter solution which uses the PMSM sensorless FOC control algorithm and complies with the IEC 60730 (UL1998) requirement, based on the TMS320F28021 device.

The goal of the FOC algorithm is to drive the motor magnetic field by generating a sine-wave current, so that the device can achieve stable output torque, high efficiency, low noise, and vibration. Sensorless FOC technology improves motor control system reliability, saves cost, and can be used for any application environment in which it is difficult to install a sensor.

The fan inverter provides rich functions including a bidirection speed controller, flying start-up, field weakening, and power calculation. The fan inverter also provides reliable protection including, OV/UV, OC, loss of phase, lock of rotor, circuit shortage, and power limitation. This solution also complies with the IEC 60730 (UL1998) standard, which is required for international home appliances.

2 System Specification

Table 1 lists the system specifications.

Table 1. System Specifications

Item		Indoor Fan	Outdoor Fan
General	Rate voltage	310 VDC	310 VDC
	Input range	250 VDC to approximately 360 VDC	250 VDC to approximately 360 VDC
	Algorithm	Sensorless FOC	
	Carry Frequency	16 kHz	16 kHz
	Current Sample	2 Shunt-resistor	3 Shunt-resistor
Motor	Rate power (Pe)	30 W	70 W
	Rate current (Ie)	0.3 A	0.6 A
	Rate speed (n)	1500 rpm	1760 rpm
	Stator resistor (Rs)	42 Ω	42 Ω
	Stator inductor (Ls)	0.137 (H)	0.32 (H)
	Pole pairs	4	4
	Type	PMSM	—
	Speed range	1000 rpm to approximately 1600 rpm	1000 rpm to approximately 1600 rpm
	Start-up rate	>95%	>95%
Performance	Noise	<36 dB	<36 dB
	Efficiency	>85%	>85%
	Fly start-up	None	<750 rpm
	Overvoltage	310 VDC*1.2 = 372 VDC	
	Undervoltage	310 VDC*0.7 = 217 VDC	
	Overcurrent	0.35 A	0.55 A
	Start-up fail	Automatic restart	Automatic restart
	Stall	Yes	Yes
Protection	Overpower	75 W	100 W
	Phase short	Yes	
	Phase miss	Yes	
Interface	Speed command	0 to approximately 3.3 V	
	Start-up voltage	>0.35 V	
	Stop voltage	<0.25 V	
	Speed feedback	12 pulses for one cycle	

Copyright © 2017, Texas Instruments Incorporated

3 MCU Use Overview

The TMS320F28021 device belongs to the C2000 Piccolo™ series with high-cost performance from TI. A brief overview of the F28021 device follows.

- 40-MHz devices, high-efficiency 32-bit CPU (TMS320C28x)
- One 3.3-V supply, no power sequencing requirement
- Integrated power-on and brownout resets
- Three 32-bit CPU timers
- 32K on-chip flash, 5K SARAM, 1K OTP memory, two internal zero-pin oscillators
- 8CH enhanced pulse width modulator (ePWM), high-resolution PWM (HRPWM)
- 13CH 12-bit analog-to-digital converter (ADC)
- 1CH enhanced capture (eCAP)
- 2CH comparator with internal DAC reference
- 1CH SCI, 1CH SPI, and 1CH I2C as communication interface
- 48-pin packages

Table 2 lists the pin assignment for the FAN inverter system.

Table 2. MCU Pin Assignment

Pin No.	Peripherals	Pin Name	Signal Name	Function
4	ADC	ADCINA6	I_Power	Bus current with filter
5, 9		ADCINA4/A2	I_Sum	Bus current
6		ADCINA7	I_U	U-phase current
7		ADCINA3	I_V	V-phase current
8		ADCINA1	I_W	W-phase current
10, 13		ADCINA0/B1	V2.48	2.48-V reference
14		ADCINB2	Vsp_In	Speed command
18		ADCINB7	V_Bus	Bus voltage
25	Capture	ECAP1	FG	Speed feedback
28	ePWM	PWM1B	UL	U-phase low IGBT driver
29		PWM1A	UH	U-phase up IGBT driver
37		PWM2A	VH	V-phase up IGBT driver
38		PWM2B	VL	V-phase low IGBT driver
39		PWM3A	WH	W-phase up IGBT driver
40		PWM3B	WL	W-phase low IGBT driver
24	GPIO	GPIO18	LED1	Run status display
26		GPIO17	LED2	Fault error display
47		TZ1	FO	Error external trigger
1	SCI	SCITXDA	TXD	UART transfer
48		SCIRXDA	RXD	UART receiver

Copyright © 2017, Texas Instruments Incorporated

4 Hardware Design

4.1 Introduction

Figure 1 shows the overall system implemented with a 3-phase HVPM motor control. The HVPM motor is driven by the conventional voltage-source inverter. The TMS320F28021 device generates the six PWM signals using a space vector PWM technique, for six power switching devices in the inverter. Two input currents of the HVPM motor (1A and 1B) are measured from the inverter and they are sent to the TMS320F28021 device using two ADCs. In addition, the DC-bus voltage in the inverter is measured and sent to the TMS320F28021 device using an ADC as well. This DC-bus voltage is necessary to calculate three phase voltages of the HVPM motor.

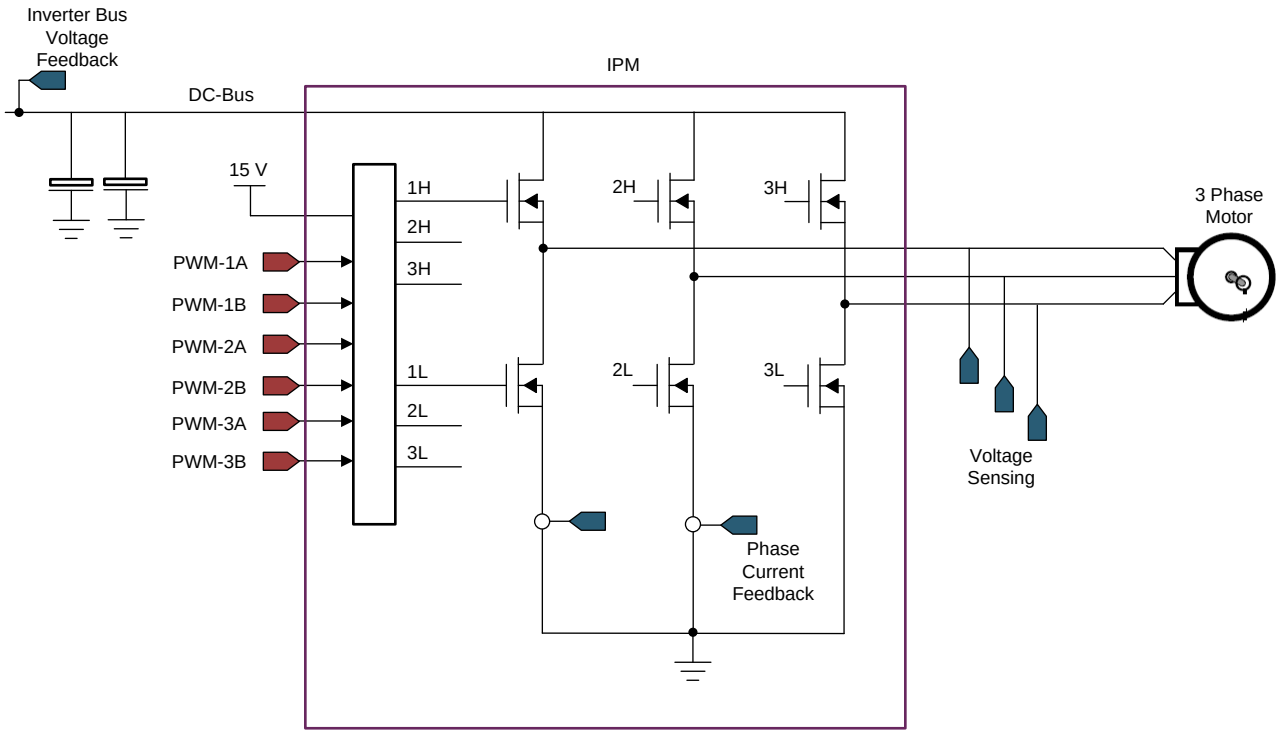

Figure 1. System Overall Structure

Copyright © 2017, Texas Instruments Incorporated

4.2 Key MCU Peripheral and Circuit Module

4.2.1 ePWM Module

One ePWM module can output two PWM signals with dead-time complementation: EPWMxA and EPWMxB. Three ePWM modules are used to generate six PWM signals: 1H/1L, 2H/2L, and 3H/3L, which drive the 3-phase power module (PS21765), as shown in Figure 2, so that the DC-BUS (300 V) can be allocated to the U-, V-, and W-phase ports of the motor. The three ePWM modules are chained together through a clock synchronization scheme which allows six PWM signals whose frequency must be aligned.

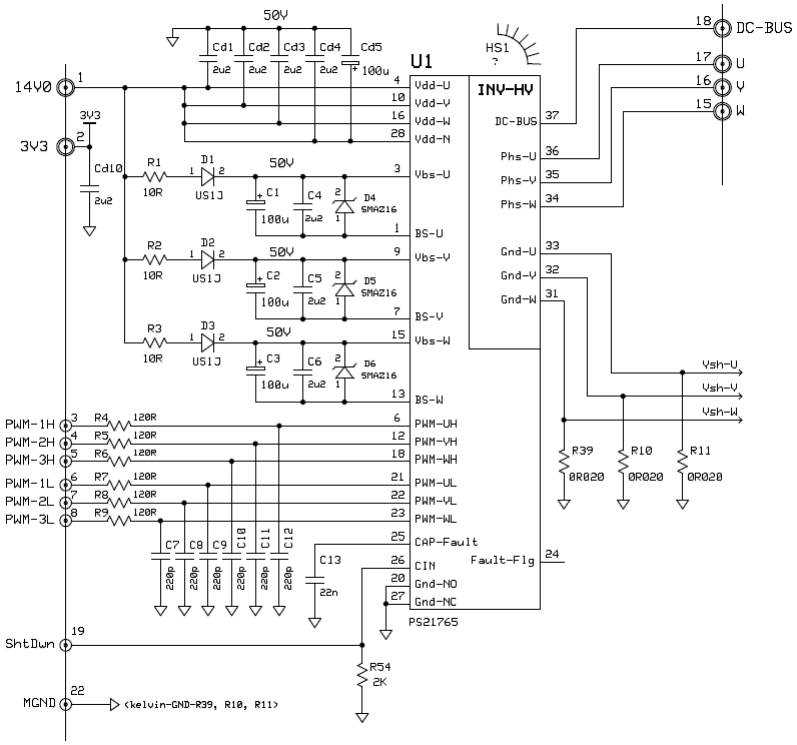

Figure 2. Three-Phase Driver Circuit

Copyright © 2017, Texas Instruments Incorporated

4.2.2 Analog-to-Digital Converter

The F28021 device has a 12-bit ADC, which contains a single 12-bit converter fed by two sample and hold circuits. The motor phase current flows through the shunt-resistor, and the voltage between the shunt-resistor indicates which motor current will be sampled. For example, the U-Phase shunt-resistor voltage (Vsh-U signal from the schematic in Figure 3) is amplified and output to the F28021 ADC as the Ifb-U signal. Beside sampling the 3-phase current, the DC bus current and bus voltage also are sampled.

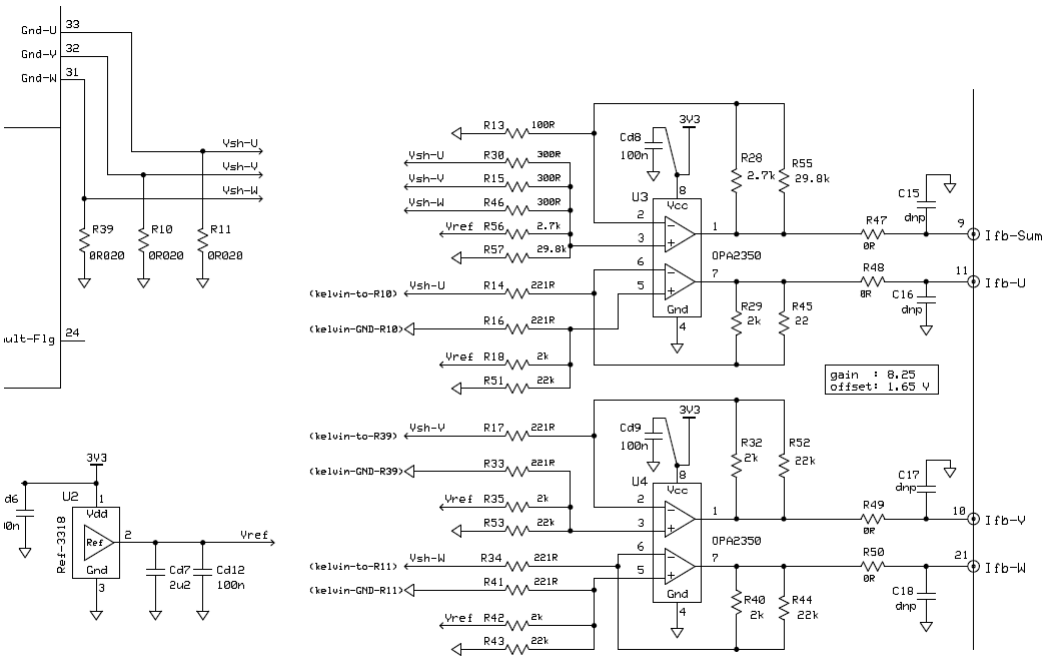

Figure 3. Current Sample Circuit

4.2.3 Comparator and Trip-Zone Block

The F28021 device has an internal analog voltage comparator. The comparator block can input one external analog signal and compare it to the internal DAC reference. The comparator output is routed to the ePWM Trip Zone modules. The 3-phase current is then input into the comparator. Once any phase current is larger than the limitation value, the comparator generates an overcurrent trigger and immediately stops the PWM output.

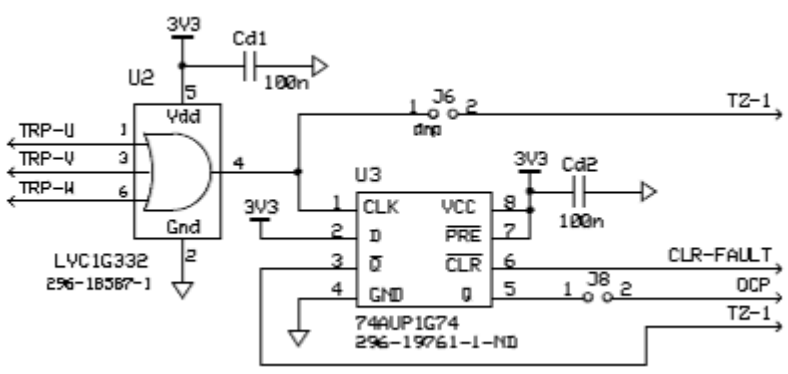

Figure 4. Comparator Protection Circuit

Copyright © 2017, Texas Instruments Incorporated

5 Firmware Design

5.1 Basic Theory for FOC

Two motor phase currents are measured. These measurements feed the Clarke transformation module. The outputs of this projection are designated isα and isβ. These two components of the current are the inputs of the Park transformation which gives the current in the d,q rotating reference frame. The isd and isq components are compared to the references isdref (the flux reference) and isqref (the torque reference). At this point, this control structure shows an interesting advantage: it can be used to control either synchronous or HVPM machines by simply changing the flux reference and obtaining rotor flux position. For a type of synchronous permanent magnet motor, the rotor flux is fixed (determined by the magnets); there is no need to create one. Therefore, when controlling a PMSM, isdref must be set to 0. Because HVPM motors need a rotor flux creation to operate, the flux reference must not be 0. This conveniently solves one of the major drawbacks of the classic control structures: portability from asynchronous to synchronous drives. The torque command isqref could be the output of the speed regulator when we use a speed FOC. The outputs of the current regulators are Vsdref and Vsqref; they are applied to the inverse Park transformation. The outputs of this projection are Vsαref and Vsβref which are the components of the stator vector voltage in the stationary orthogonal reference frame. These are the inputs of the Space Vector PWM. The outputs of this block are the signals that drive the inverter.

NOTE: Both Park and inverse Park transformations need the rotor flux position. A sliding-mode-based estimator algorithm is used to calculate the rotor flux position by analyzing the sampled motor current and voltage.

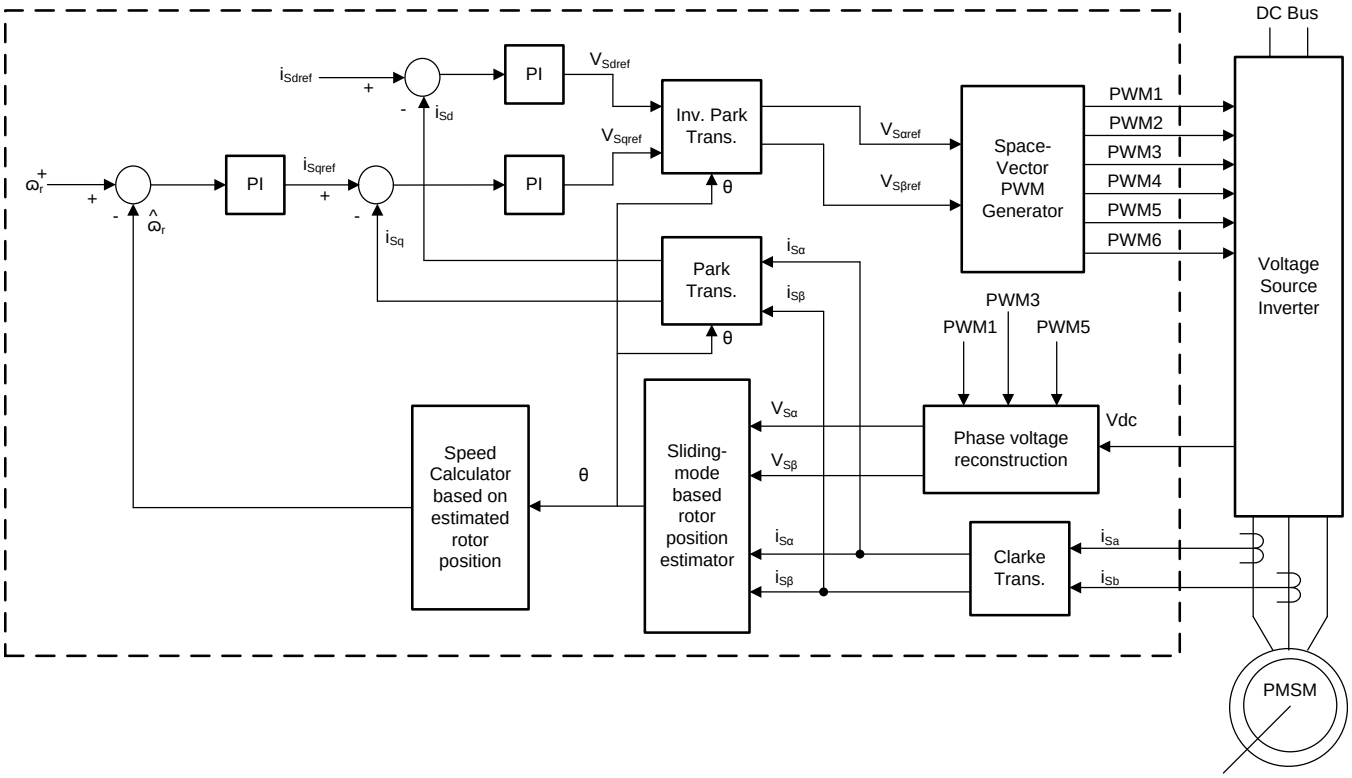

Figure 5. FOC Algorithm Structure

Copyright © 2017, Texas Instruments Incorporated

5.2 Software Process Flow

Figure 6 shows the software process flow chart definition for this motor control system.

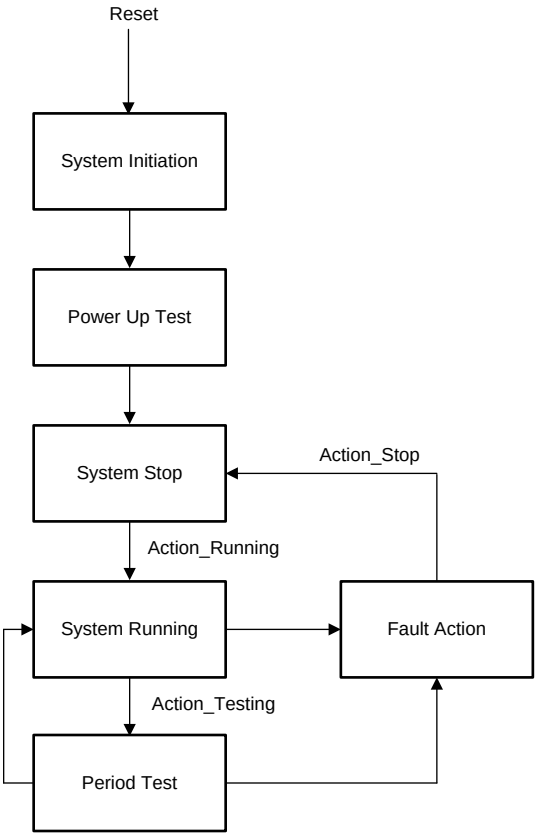

Figure 6. Main Procedure Flow

Copyright © 2017, Texas Instruments Incorporated

5.3 Software Design Action Functions

5.3.1 System Initiation

This step is executed in the beginning to configure all parameters and device peripheral modules, which includes the motor parameter, user interface definition, and peripheral settings for the ADC module, PWM module, and GPIO module.

5.3.2 Power-Up Testing Sequence

This step is executed at power-up or reset before running the system. If the power-up test fails, then the software stops system from running.

The power-up test includes:

- Device ID detection
- Stack corruption detection
- Clock fail detection
- Flash CRC detection

5.3.3 Running the System

This step is executed while the motor control algorithm is running. The motor goes into the park state, then into start-up by the open-loop control, and lastly runs on the closed-loop state, in which the motor is driven to command setting speed.

5.3.4 Period Self-Test Initiation and Enabling

This step must be executed by the control algorithm after the motor is already running. The periodic test cycle is enabled by set the software flag, then the periodic test always runs at the background process. The periodic test monitors the system fault and microelectronic fault, and sets the fault flag when the test function detects abnormal behavior in the control IC or motor.

5.3.5 Stopping the System

This step stops the system and resets the parameter by defining the sequence flow. The step can be triggered by the user control command or when an unwanted fault is detected, to protect the system from sustaining more damage.

5.3.6 Fault Action State

This step immediately disables the PWM driver signal when a defined fault is detected. For example, one of the predefined limits is exceeded. In this case, the motor shuts down and the software generates a fault flag. The fault type can be read in the Fault Flags. Multiple faults can be present at the same time, and they are all handled with the same priority. The motor can be restarted only after the fault condition has been removed (fault monitoring continues after stopping the motor).

Copyright © 2017, Texas Instruments Incorporated

6 Fault Handle

6.1 Introduction

For the motor control system, some critical faults can occur before the system starts or while the system is running. Therefore, the control system must monitor the fault condition to avoid any injury to users and protect the system from harm. When any fault condition occurs, an interrupt is generated from the MCU and the fault is reported by setting the fault flags. During this time, the MCU immediately shuts down the system. When the motor has been stopped due to a fault condition, it cannot be restarted until the fault is cleared. To clear a fault condition, write 0 to the fault flags.

6.2 Key Risk and Fault Analysis

6.2.1 Motor Running Heating

During the motor running process, if the incorrect current is added to the motor driving phases, this energy causes motor heating. Overcurrent protection circuitry is required in most applications to prevent heat damage to the hardware and the motor.

Overcurrent protection includes two parts: hardware protection and software protection.

Hardware protection is the fastest acting protection function. It turns off all the PWM signals using the MCU-integrated analog comparator when it detects an overcurrent event. Both the DC current and SUM of the 3-phase current are monitored, if either is over the threshold value, then the analog comparator inside the MCU generates the OST signal to turn off all PWM signals by hardware logic.

Software protection is the slower current protection function. The current is sampled by the ADC and then calculated using the firmware code. If the firmware determines that the current is over the defined limit, the firmware turns off the PWM signals and sets the fault flag. Figure 7 shows the current sample circuit and firmware procedure flow chart.

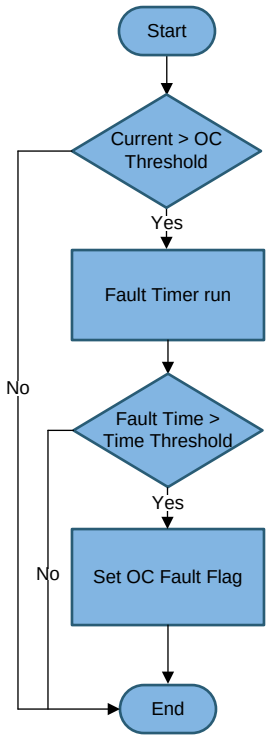

Figure 7. Overcurrent Protection Flow

Copyright © 2017, Texas Instruments Incorporated

6.2.2 Motor Loss of Phase

There are two possible conditions for the loss of phase behavior: motor stop and motor running. Both conditions are monitored and protected using two different approaches.

During the motor stop condition, the firmware checks whether phase loss or short circuiting occurred using addition driver signals. The firmware detects response current signals and the procedure calls the *FAN_CHK_CTRL* state. Figure 8 shows the flow chart description.

During the motor running condition, the system checks for phase loss or short circuiting by detecting the DC overcurrent signal. The method is similar to that in Section 6.2.1.

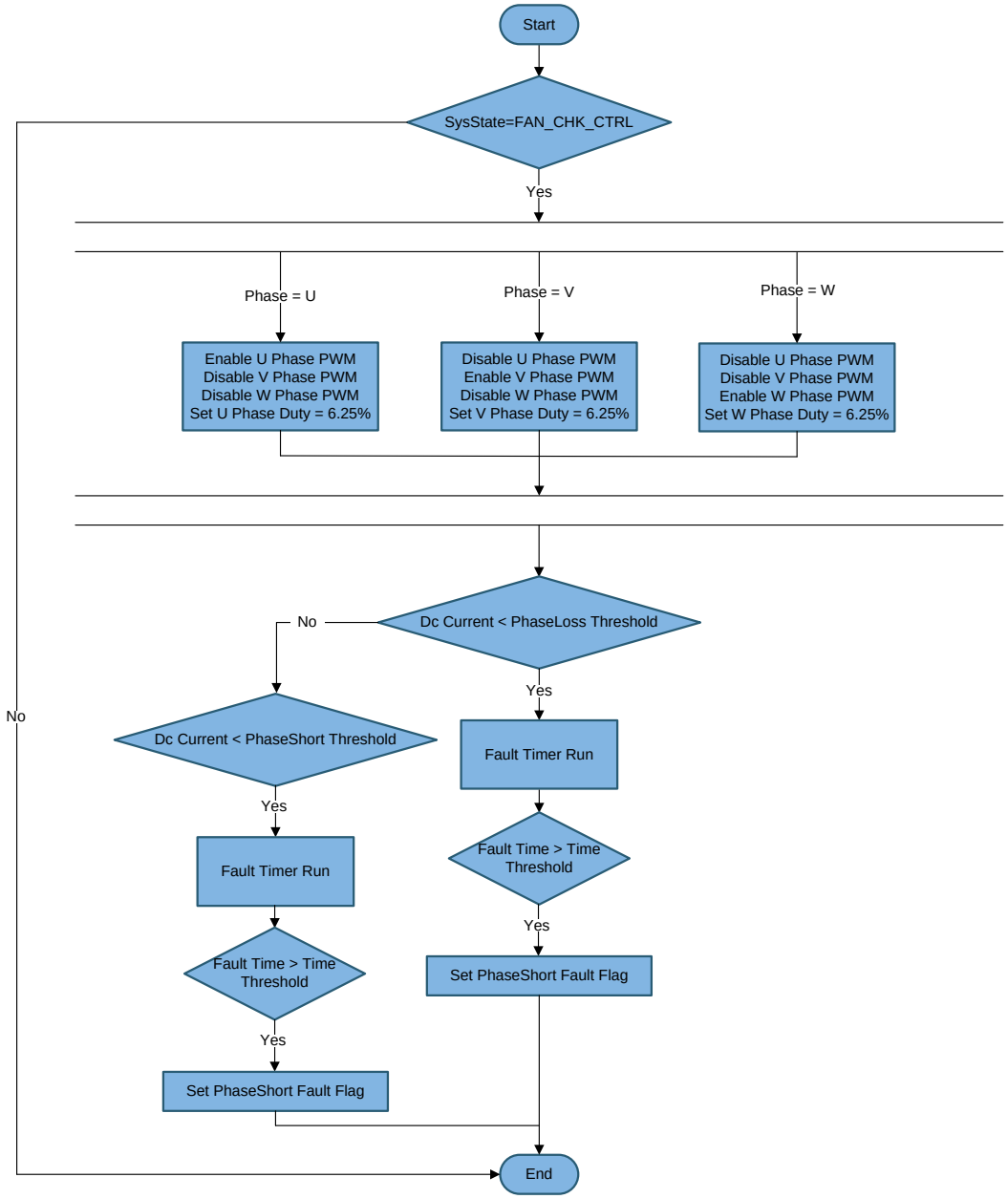

Figure 8. Loss Phase Protection Flow

Copyright © 2017, Texas Instruments Incorporated

6.2.3 Lock of Rotor

The motor lock occurs when an abnormal current waveform is detected. The difference detect threshold must be used for the motor stop or running states. Figure 9 describes the firmware procedure.

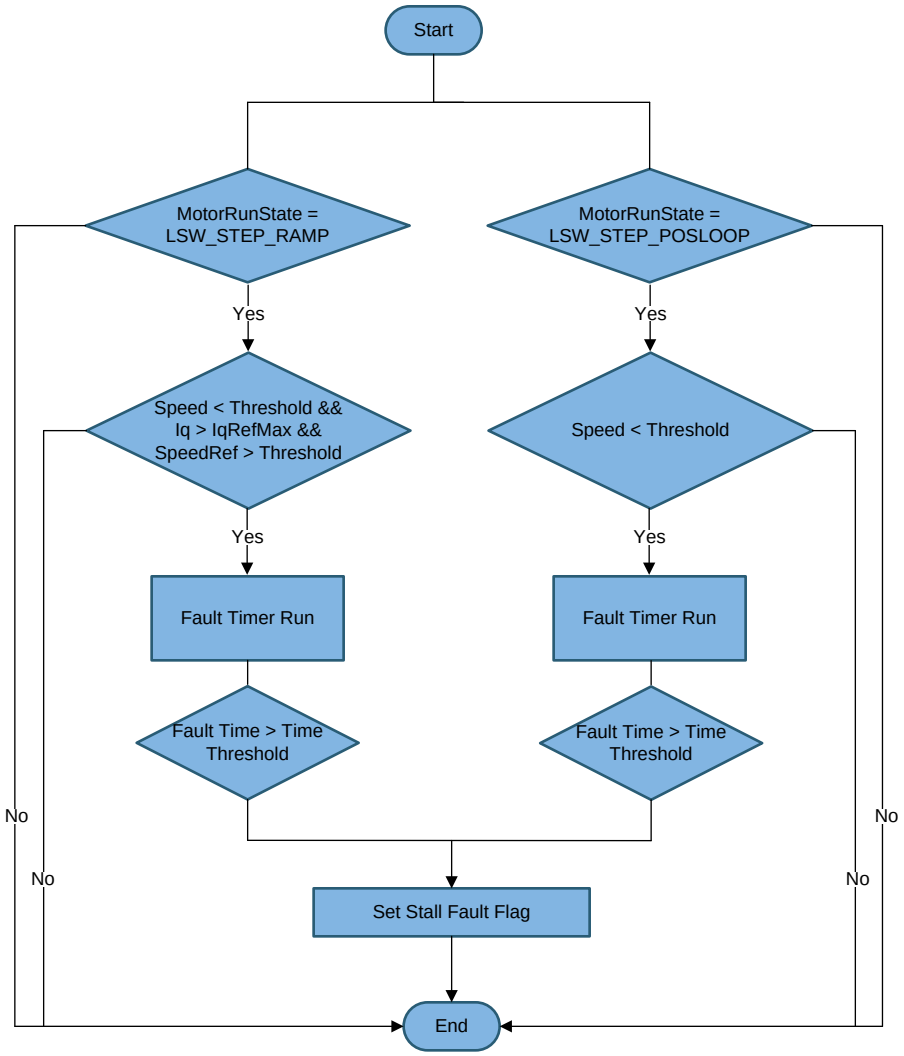

Figure 9. Lock of Rotor Protection Flow

6.3 Common Failures and Protection Mode

6.3.1 Overcurrent

Overcurrent protection circuit prevents damage to the motor and inverter by shutting down the PWM switching outputs of the control IC when the current across the low-arm shunt resistor reaches a threshold level. When verifying new hardware, the overcurrent protection circuit must be tested as the first priority in the process.

6.3.2 Overvoltage

An overvoltage fault is detected when the DC bus voltage exceeds the established threshold level. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The fault states restore when the DC bus voltage returns to normal range.

Copyright © 2017, Texas Instruments Incorporated

6.3.3 Undervoltage

An undervoltage fault is detected when the DC bus voltage falls beneath the established threshold level. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The fault states restore when the DC bus voltage returns to normal range.

6.3.4 Start-Up Fail

If the actual speed of the motor is far from the command speed, then it reaches the threshold level when switching from open-loop to closed-loop, and a start-up fault is detected. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The system tries to restart the motor until the motor can switch to closed-loop.

6.3.5 Stall Fail

This function checks if the motor stops rotating and is executed during both start-up and normal running. If for some reason the rotor stops rotating and the motor speed falls below the threshold level, a stall fault is detected. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The system tries to restart the motor until it can run normally.

6.3.6 Current Limitation

This function checks if the motor reaches the threshold level during each phase current. It the motor reaches the threshold level, then a current limitation fault is detected. If detected, the system prevents damage to the motor and inverter by decreasing the motor speed to keep the phase current at or below the threshold level. The system implements the speed limitation until the current returns to normal range.

6.3.7 Power Limitation

This function checks if the motor power consumption reaches the threshold level. If the power consumption reaches the threshold, then a power limitation fault is detected. If detected, the system prevents damage to the motor and inverter by decreasing the motor speed to keep the power consumption at or below the threshold level. The system implements the speed limitation until the power consumption returns to normal range.

6.3.8 Phase Loss

This function checks for phase loss, and it is executed during both during start-up and normal running. During start-up, each phase current is checked for whether it is equal to the command current. If the current error is greater than the threshold level, a phase loss fault is generated.

During normal running, each phase rms current is checked for balance. If the current unbalance is greater than the threshold level, then a phase loss fault is generated. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The system tries to restart the motor until the motor can switch to closed-loop.

6.3.9 Overtemperature

An overtemperature fault is detected when the IPM temperature sensor signal reaches the established threshold level. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The fault states restore when the IPM temperature sensor signal returns to normal range.

6.3.10 Over Speed

If the motor command speed exceeds the established threshold level, an over-speed fault is detected. If detected, the system prevents damage to the motor and inverter by shutting down the PWM switching output of the control IC. The fault states restore when the command speed returns to normal range.

Copyright © 2017, Texas Instruments Incorporated

7 Microelectronic Fault Protection Function

To simulate the microelectronic failure, users must rebuild the program, enabling the corresponding macro definition.

For each rebuild, only one error simulation macro can be enabled by setting it to 1. After rebuilding and programming the MCU, the gErrorTestFlag flag is used to start the error condition.

The error macro is defined in two files: STL_system_config.h (see Figure 10) and STL_test_report.h (see Figure 11).

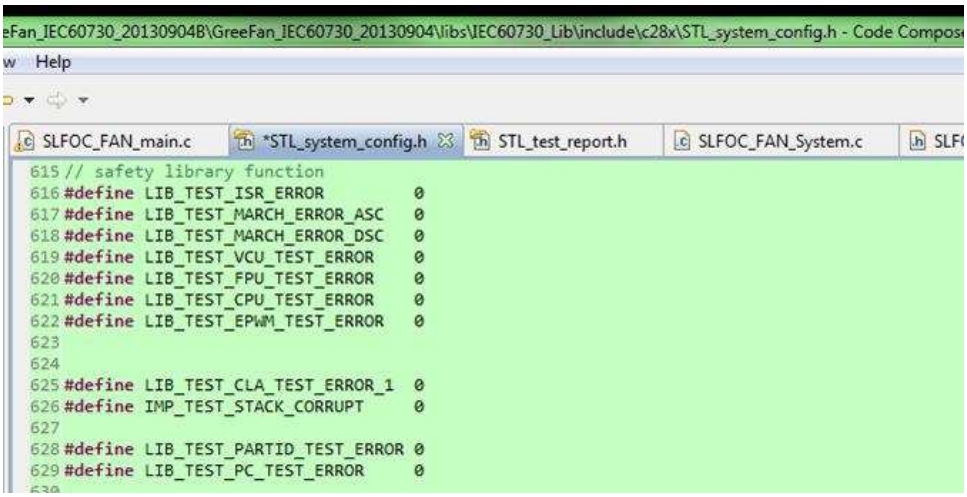

Figure 10. System Config Header File

When the macro in the STL_system_config.h file is changed, the IEC 60730 lib project must be rebuilt too. After the lib rebuild, application projects such as the Fan_Motor project must be rebuilt with the new .lib file.

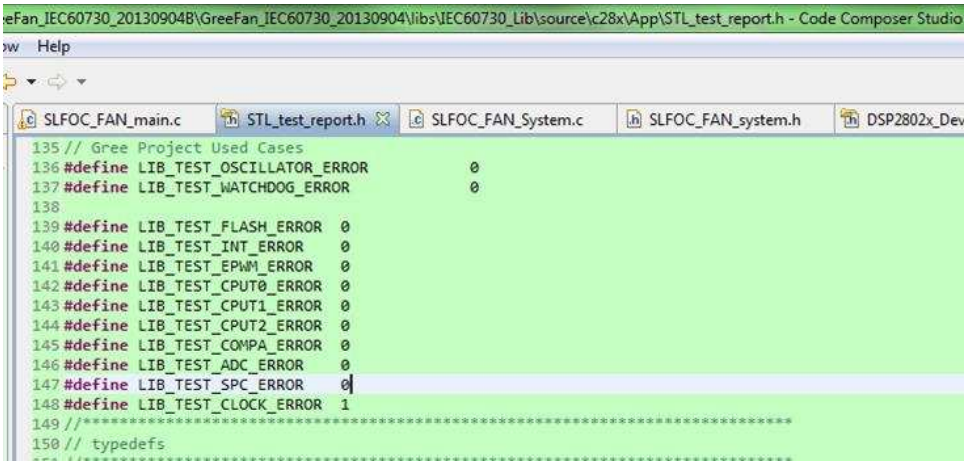

Figure 11. Error Simulation Header File

7.1 Part ID

This method checks the MCU silicon ID number, which is stored in the dedicated silicon memory. This check only runs one time when the board is powered up. If the check fails, a related fault code is set (the designer can watch this code in the debug environment).

To simulate the error, ensure the test code configures the wrong part number and start the test.

Copyright © 2017, Texas Instruments Incorporated

7.2 CPU Test

This function tests the CPU core registers for stuck bits periodically. The following registers are tested:

- ACC
- P
- XAR0 to XAR7
- XT
- SP
- IFR, IER, and DBGIER
- ST0
- DP

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: set a different code pattern into the registers and start the tests.

7.3 PC_TEST

This function tests the program counter register for stuck bits.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong program counter address before the test start.

7.4 RAM_TEST

RAM test include the following:

- Program RAM test
- Data RAM test
- SafeRam Stack Test
- SafeRam Boot Test
- SafeRam PsaCrc test
- SafeRam Pie vector test
- SafeRam PC test

The test writes and reads the RAM area periodically to verify the RAM functionality.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong program counter address before the test starts.

7.5 FALSH_TEST

This function calculates the CRC value of the used flash area and checks the stored CRC value periodically.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong golden code before the test starts.

7.6 SPC_TEST

This function is the MCU hardware stack watch point, which is configured to supervise the stack corruption. If a stack corruption occurs, the SPC interrupt is generated.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Copyright © 2017, Texas Instruments Incorporated

Error simulation: as the hardware watch point is configured, the SPC error occurs when the RAM stack test executes. Normally, when the RAM stack test is processing, the SPC watch point function interrupt service routine (ISR) is ignored. For the SPC error simulation, do not ignore the stack watch point corruption ISR.

7.7 INTERRUPT_TEST

Use the watchdog clock timer to check if the interrupt service function can be executed at the right frequency.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong watch dog timer threshold before the test starts.

7.8 OSCILLATOR_TEST

This function tests whether the internal oscillator runs at the wrong frequency of 10 MHz.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong Program counter address before the test starts.

7.9 ADC_TEST

This function tests the ADC module by measuring the ADC A0 and B1, which is linked with a 2.48-V constant signal, to verify the ADC functionality.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong ADC check threshold value before the test starts.

7.10 COM_TEST

The overcurrent ADC signal is connected with two comparator modules of the MCU. The test function checks the output of the two compare modules. If the output of the modules is different the test fails; otherwise, the test passes.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong comparator DVAL value to generate the wrong compare status for comparator 1 before the test start.

7.11 WATCHDOG_TEST

This function uses the watchdog overflow interrupt to count the watchdog period count, to verify the watchdog functionality.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong delay time before the test starts.

7.12 TIMER_TEST

This function uses the watchdog timer count to tests CpuTimer0, CpuTimer1, and CpuTimer2.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong watchdog timer threshold before the test starts.

Copyright © 2017, Texas Instruments Incorporated

7.13 EPWM_TEST

This function uses the watchdog timer count to test the ePWM module period accuracy and functionality.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force a wrong watchdog timer threshold before the test starts.

7.14 CLOCK_FAIL_DECTECT

This function uses the hardware monitor to supervise the clock missing issue. If a clock missing event occured, an NMI interrupt is generated.

If the check fails, a related fault code is set (the designer can watch this code in the debug environment), and the motor is stopped.

Error simulation: force the NMI clock fail flag to generate the clock fail event.

8 Final Product Performance

8.1 Power Consumption

Figure 12 shows the motor power consumption from minimum speed (600 rpm) to maximum speed (1600 rpm). The power consumption fits the design specification.

电流/A	功率/W	转速/rpm
0.097	8.3	600
0.111	10.5	700
0.135	13.2	800
0.18	17.5	900
0.224	22.6	1000
0.278	28.1	1100
0.322	35.1	1200
0.388	42.6	1300
0.484	52.1	1400
0.558	62.2	1500
0.682	74.1	1600

Figure 12. Power Consumption Record

Copyright © 2017, Texas Instruments Incorporated

8.2 Motor Running Waveform

Figure 13 shows the motor phase current waveform for around 60 Hz, and Figure 14 shows 100 Hz speed. The waveform is smooth and stable.

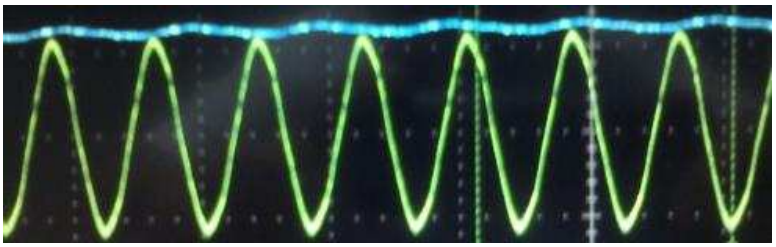

Figure 13. 60-Hz Current Waveform

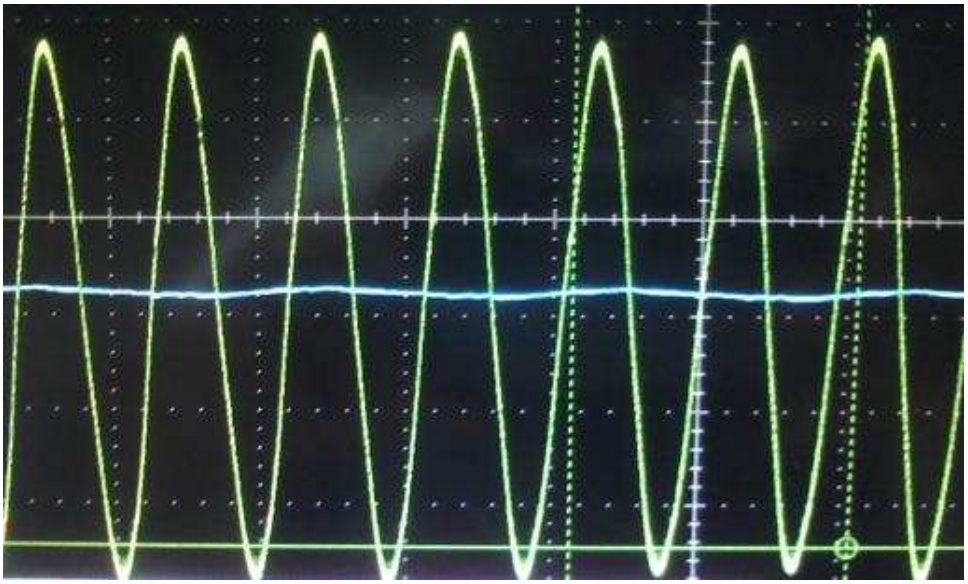

Figure 14. 100-Hz Current Waveform

Copyright © 2017, Texas Instruments Incorporated

9 Related Documentation

1. Texas Instruments, *TMS320F2802x*, Data Sheet
2. Texas Instruments, *TMS320F2802x*, Technical Reference Manual
3. Texas Instruments, Sensorless FOC of PMSM, C2000 Systems and Applications
4. Texas Instruments, IEC60730_F2803x_STL_User_Manual

Copyright © 2017, Texas Instruments Incorporated

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2017, Texas Instruments Incorporated