

To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: <http://www.renesas.com>

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (<http://www.renesas.com>)

Send any inquiries to <http://www.renesas.com/inquiry>.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
 - “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
 - “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
 - “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

SH2/2A Family

Sensorless Vector Control of Three-Phase Permanent Magnet Synchronous Motors

Introduction

This application note presents Renesas' sensorless vector control solution based on the SH2-7086 microcontroller (MCU), which is part of Renesas' the SH2/2A Family, to enable high performance and cost effective control for permanent magnetic synchronous motors (PMSM). It describes the implementation hardware platform, the SH7086 MCU, software design, and sensorless vector control theory. Additionally, the document explains specifically how to modify the solution for different motors and applications. An example of parameter tuning is given.

Target Devices

SH2/2A Family

Contents

1. Overview	4
2. Sensorless Vector Control Features	4
3. Specification and Performance	5
4. Concept of Sensorless Vector Control of PMSM Motors	5
4.1 PMSM Motors	5
4.2 Sensorless Vector Control	5
5. Implementation Hardware Platform	7
6. SH7086 Microcontroller	8
7. SVC Software Organization	9
7.1 Overall Software Architecture	9
7.2 Core Modules	10
7.2.1 Initialization	10
7.2.2 Parameter Definitions	10
7.2.3 AD Sampling	11
7.2.4 Speed Setting and Ramp Generation	11
7.2.5 Startup Procedure	12
7.2.6 Speed and Current PI Regulator	12
7.2.7 PWM generation	13
7.2.8 Protections	13
7.3 Modules in SVC Library	13
7.3.1 Clarke, Inverse Clarke, Park and Inverse Park Transformations	13
7.3.2 PWM Duty Calculation	14
7.3.3 Flux and Speed Estimation	14
8. Software Descriptions	15
8.1 SVC Software Workspace	15
8.2 Main Code	16
8.3 Motor Control Code	16
8.4 SVC Library Code	17
8.5 Flowchart of PWM Interrupt	18
9. Motor and Control Parameters	19
9.1 Parameters in customize.h	19
9.2 Operation Variables	22
9.3 Parameter Tuning Example	22
10. MCU Performance Analysis	24
11. Vector Control Theory	24
11.1 Space vector definition	25
11.2 Clarke Transformation	25
11.3 Park Transformation	26
11.4 Inverse Park Transformation	27

11.5 Inverse Clarke Transformation.....	27
12. Modeling Vector Control of PMSM Motors.....	28
13. Sensorless Vector Control Methodology.....	29
13.1 Flux and Position Estimation.....	29
13.2 Speed Estimation	30
14. References.....	30
Website and Support.....	30
Revision Record	30

1. Overview

Vector control (VC) has been widely used in various industrial applications. It is one of the most popular schemes for high performance of three-phase AC motor drives. Through the coordinate transformation, VC decouples three phase stator currents into two phase dq-axis currents, one producing flux and the other producing torque. This allows direct control of the flux and torque, in the same manner as for DC motors, and thus achieves fast dynamic response and high performance.

In recent years, there has been an increase in sensorless vector control (SVC) for permanent magnetic synchronous motor drives (PMSM). This is primarily due to availability of high-performance and low-cost microcontroller (MCU), which makes it possible to implement more sophisticated motor control techniques, such as SVC, through software, and thus to achieve fully digital and high-performance motor control systems. SVC eliminates speed sensor, and both motor speed and flux are directly acquired through observer or estimator after processing information of terminal currents. This not only reduces cost of the drive system, but also improves system reliability, which is crucial for many applications.

This application note presents Renesas' sensorless vector control technology and implementation of 3-phase permanent magnetic synchronous motor based on Renesas' SH7086 microcontroller.

Software described in the application note is applicable to following devices and platforms.

- MCU: SH7086 or SH2/2A family
- Motor: three-phase BLDC and PMSM motors
- Platform: Renesas' MCRP-Shakti II power board
- Control algorithm: sensorless vector control

2. Sensorless Vector Control Features

Renesas' sensorless vector control adopts a unique implementation strategy. Instead of direct integration of back EMF, the flux is estimated through two low-pass filters.

It has the following features:

- Parameter insensitive
- Eliminating integration error and DC offset
- No position estimation offset
- No initial position and flux requirements
- Simple to implement and little computation involved
- Easy to modify for different motors
- Direct flux and torque control
- Sinusoidal motor phase currents
- High startup torque (80% of full load)
- Good speed regulation ($\pm 10\text{rpm}@500\text{rpm}$)
- Fast dynamic response ($<1\text{s}$ with full step load)
- Very stable and reliable
- Good dynamic and steady state performance
- High efficiency
- Low noise

3. Specification and Performance

The implementation of SVC is based on Renesas' MCRP-Shakti II power platform and SH7086 MCU, the main specification data are described as following:

- Input voltage: 80 ~250VAC
- Input frequency: 40 ~70Hz
- Rated bus voltage: 320V
- Rated output power: 2kW
- Switch frequency of inverter: 20KHz
- Speed rang: above 500rpm for 1 pair of poles of motors
- Current measurement: LEM sensors
- Used flash memory: 13Kbytes
- Used RAM: 700bytes
- Used Stack : 168bytes

4. Concept of Sensorless Vector Control of PMSM Motors

4.1 PMSM Motors

The advents of high performance magnets, such as samarium cobalt and neodymium boron iron, have made it possible for permanent magnet motor drives to achieve performances that can surpass conventional DC motors and induction motors, and thus are becoming more and more attractive to industrial applications. In comparison with induction motors, PM motors have distinct advantages, including high power density, high torque to inertia ratio, high efficiency, and better controllability.

Permanent magnet motor drives can be categorized into two types. The first category, BLDC motor drive, is based on position that is not continuous but at fixed points, typically every 60 electrical degrees for commutation of phase currents. The ideal back EMF is trapezoidal. Correspondingly, BLDC motors adopt six-step trapezoidal control. While the control is simple, easily implemented, and low cost, it can cause torque ripple and noise. Also the performance and efficiency are not high. The second category is called PMSM motor drive, which uses continuous rotor position feedback with sinusoidal voltages and currents generated by PWM modulation of DC bus. Because the ideal back EMF of a PMSM motor is sinusoidal, constant torque with very low ripple is produced. This improves torque ripple, noise and efficiency. Additionally, PMSM motors are suitable for advanced motor control methods like vector control.

4.2 Sensorless Vector Control

For vector control, a three-phase space voltage is generated and used as a vector to control three-phase stator currents. By transforming physical phase currents into a rotational vector using Clarke and Park transformations, the flux and torque currents become time-invariant, allowing direct control of the motor flux and torque, as for DC motors, and thus achieving fast dynamic response and high performance. By eliminating speed sensor, SVC directly acquires the flux and speed, etc. through observer or estimator after processing the information of terminal voltages and currents. It not only reduces cost of the drive system, but also improves system reliability.

Figure 1 shows a block diagram of three-phase motor control system. The input is a single phase AC power supply -110 volts or 220 volts, which is converted into 160V or 320V DC voltage. The three-phase inverter adopts voltage source inverter (VSI) and uses power modules of 10A, 16A or 20A, which generates three phase voltages with variable frequency and amplitude to drive the motor to the desired voltage. The PWM pattern for the power module is controlled by SH7086 MCU using sensorless vector control technology.

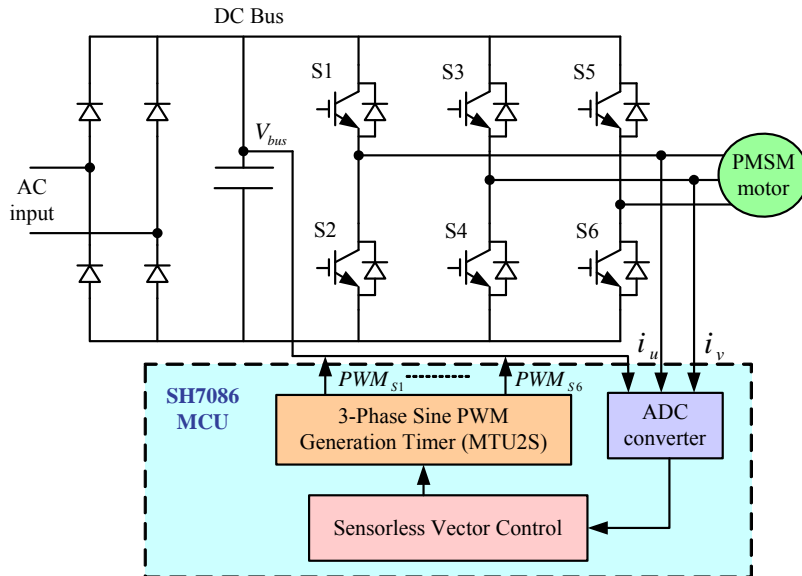


Figure 1 Block diagram of SVC control system

Depicted in Figure 2 is a block diagram of SVC, showing functions required for SVC, including speed control loop, d and q current loops, vector control frame transformations (uvw to $\alpha\beta$ Clarke transformation, and $\alpha\beta$ to dq park transformation), and speed and position estimation. The software implementation is based on this block diagram.

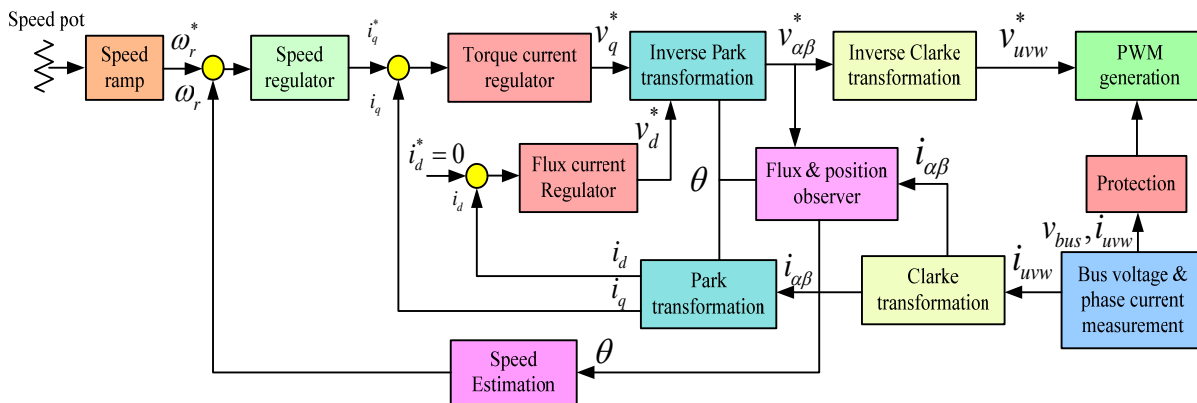


Figure 2 Block diagram of SVC of three-phase PMSM motors

The outer control loop is speed control loop. The commanded speed ω_r^* is input from an external potentiometer, and compares it with motor actual speed ω_r , that is observed from motor terminal currents without speed sensor. The speed PI controller uses the conventional PI controller. Its output is the required torque current i_q^* .

The motor currents of i_u, i_v, i_w are measured by three LEM current sensors. For a balanced three-phase system, it is only necessary to measure two phase currents of i_u, i_v , and the third current can be calculated. The motor actual currents are transformed into i_α, i_β and i_d, i_q currents by Clarke and Park transformation. There is no position or speed sensor. The motor phase currents of i_α, i_β and voltages of v_α, v_β are used to estimate rotor position and motor speed.

The inner control loops are the d-axis current i_d and q-axis current i_q control loops. Because the BLDC motor is a permanent magnet motor, the flux is generated by the mounted permanent magnet in the rotor. The current is not

necessary for producing the flux, and thus i_d is set to zero. The current PI loop outputs are the d and q voltages of v_d, v_q , which are transformed back into the motor phase voltages of v_u, v_v, v_w by the inverse Clarke and Park transformations so that six PWM signals are generated to drive the motor to the desired voltage.

5. Implementation Hardware Platform

Renesas' SVC is implemented with Renesas' MCRP-Shakti II power board and SH7086 RSK. The functional block diagram is shown in Figure 3. The system is versatile and can be applied to any types of motors driven by a three-phase power inverter. The voltage source inverter (VSI) is used to regulate the motor speed of three-phase PMSM motors by varying frequency and voltage.

The system consists of an input AC/DC rectifier, a DC link and an output DC/AC inverter. It is capable of driving high voltage medium current motors. The bus voltage could be up to 400 volts with the current up to 20 amps. The board is designed to measure bus voltage, bus current, and three phase motor currents through LEM current sensors. The three phase motor back EMFs are also detected by ADC converters or three comparator circuits. The speed is input through hall sensor and encoder circuits. The centre processing unit (CPU) can use any Renesas' microcontroller RSKs, such as: R8C, M16C and SH2/2A, which are directly plugged into the board.

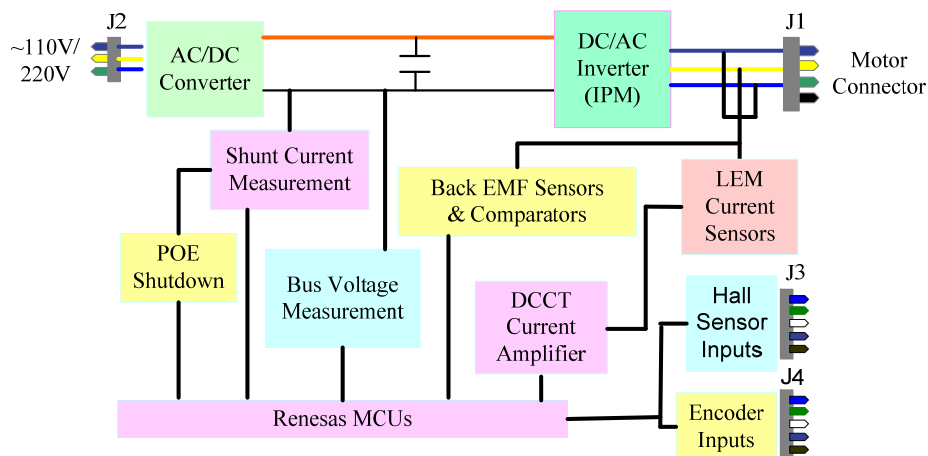


Figure 3 Functional Block diagram of MCRP-Shakti II power board

Shown in Figure 4 is a MCRP-Shakti II power board, which has following features:

- Speed sensor inputs of hall sensor, encoder, and tachometer;
- Two high side phase current measurement via LEMs and one shunt current measurement using precision resistor and special timers;
- 110 or 220VAC input and output, and bus voltage of 160V or 320V;
- SH7086 three-phase timer supports multiple PWM modes including complementary pairs with automatic dead-time insertion;
- Support various RSKs of R8C, M16C and SH2/2A series to directly plug in;
- Various power modules can be used (10A, 16A and 20A modules).

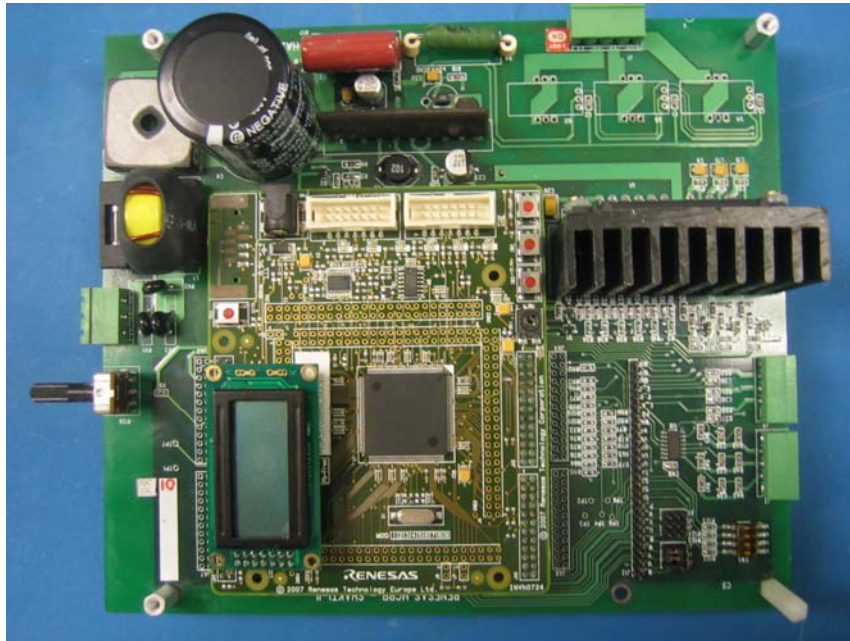


Figure 4 MCRP-Shakti II power platform with SH7086 RSK

6. SH7086 Microcontroller

SH7086 is a 32-bit and 80MHz microcontroller. As part of SH Family of microcontrollers, SH7086 MCU offers high performance and high throughput required for today’s sophisticated designs, especially vector control and other high performance motor control methods. Figure 5 is a block diagram of SH7086 MCU.

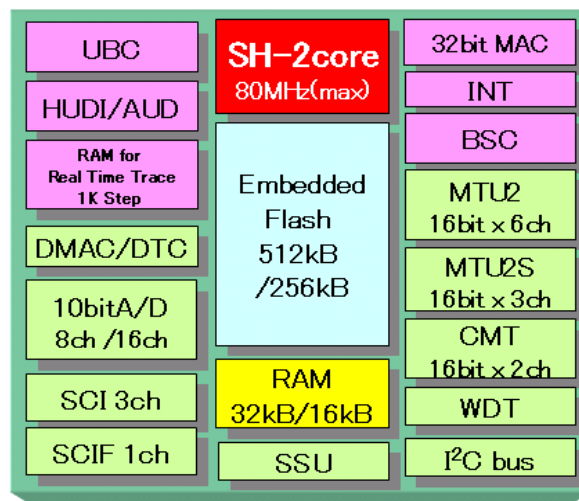


Figure 5 Block diagram of SH7086 MCU

Key features of SH7086 MCU:

- High-performance single-chip RISC with SH-2 core
 - 104MIPS at 80MHz
 - Built-in 32-bit multiplier
- Built-in large-capacity memory (ROM/RAM)
 - Flash memory of 512KB
 - RAM of 32KB
- Peripheral functions
 - Powerful timer: MTU2(16bit×6ch), MTU2S(16bit×3ch), Compare match timer (CMT)(16bit×2ch)

- A/D converters (10-bit resolution) :
10bit, 16ch (4ch x 2unit + 8ch x 1unit)
- Serial IO with FIFO (SIOF) × 1channel
- Serial communication interface (SCI) × 3 channels
- Synchronous serial communication unit (SSU) × 1 Channel
- I2C Bus Interface : 1ch (except SH7083)
- DMAC: 4 channel
- Data transfer controller (DTC)
- Bus state controller (BSC): Connectable with various types of memory such as SDRAM.
- On chip WDT ×1 channel
- H-UDI: On-chip debugging functions

7. SVC Software Organization

7.1 Overall Software Architecture

Renesas' SVC algorithm for three-phase PMSM motor drives is implemented with the complete C code. The overall software architecture is shown in Figure 6. It is composed of two main blocks, SVC library and SVC core modules. SVC library includes vector control transformations (Clarke and Park transformation), PI controller, flux and speed observer, and PWM duty calculation. Once the SVC algorithm is developed, the modules in this block can be directly applied to different motors and control systems without any changes. The blocks in SVC core modules are motor and MCU initialization, parameter definitions, ADC sampling, speed setting and ramp generation, startup procedure, current id and iq regulators, speed regulator, PWM generation and fault protections. Following is a brief description of each of core modules.

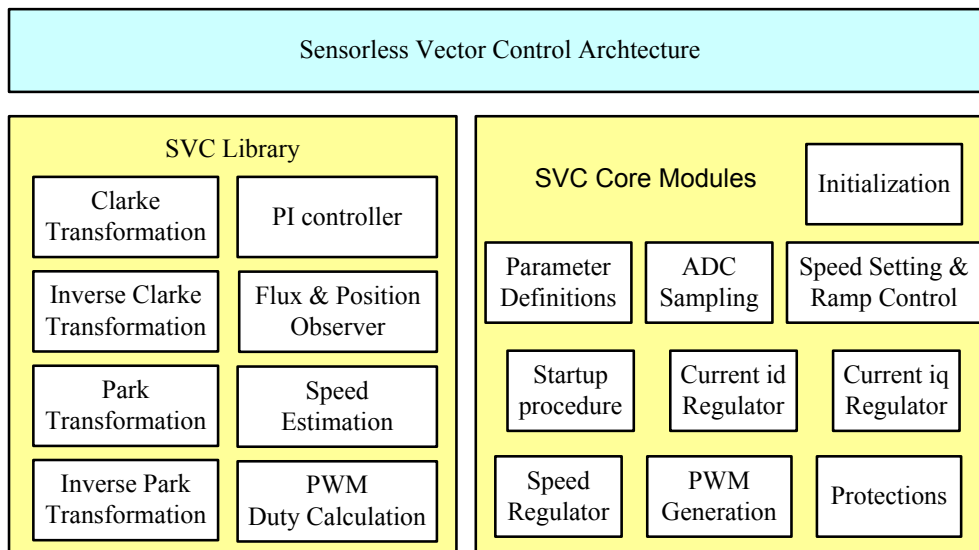


Figure 6 SVC software architecture

7.2 Core Modules

7.2.1 Initialization

The initialization module includes definitions of MCU registers such as MTU2S for three phase PWM outputs, motor and control parameters, LCD display definitions, and motor current offset calculation. Below is a list of functions in this module.

- HardwareSetup(): defines SH7086 MCU registers related with SVC such as: CPG, STB, PE, INT, ADC, MTU2S.
- SVC_Init_Par(): initializes variables of motor and control parameters.
- SVC_Flux_Est_Par(): initializes flux estimation filter parameters.
- SVC_Speed_Est_Par(): initializes speed estimation filter parameters.
- SVC_Init_PWM(): sets up MTU2S for three phase PWM outputs.
- SVC_Start_PWM(): starts PWM and PWM interrupt.
- SVC_PWM_Start_Out(): enables PWM outputs.
- Current_Offset(): calculates current sensor offsets.

7.2.2 Parameter Definitions

A header file “customize.h” contains initial definitions of motor and control parameters, including:

- Motor parameters;
- ADCs scaling – motor phase currents and DC bus voltage;
- Startup acceleration ramp;
- Speed and current PI gains;
- Flux estimation filter parameters;
- Speed estimation filter parameters ;
- Current sensor selection: one shunt, two shunts, or LEM current sensors;
- PWM carrier frequency – 10KHz or 20KHz.

The detailed definitions of these parameters are described in next section. Here is an example of definitions of motor parameters, current and speed PI gains.

```

• #define      RPM_MIN_CUSTOM          100    // minimum motor speed in rpm
• #define      RPM_MAX_CUSTOM          2500   // maximum motor speed in rpm
• #define      R_ACC_CUSTOM            100    // acceleration ramp in rpm/sec
• #define      POLES_CUSTOM            2      // polar pair number of motor
• #define      I_START_CUSTOM          15     // startup current in Amps/10
• #define      IQ_MAX_CUSTOM           45     // maximum iq current in Amps/10
• #define      R_STA_CUSTOM            54     // stator phase resistance in Ohm/10
• #define      KP_CUR_CUSTOM           200    // proportional gain of current controller
• #define      KI_CUR_CUSTOM           100    // integral gain of current controller
• #define      KP_SPD_CUSTOM           300    // proportional gain of speed controller
• #define      KI_SPD_CUSTOM           200    // integral gain of speed controller

```

For Renesas’ SVC, there are only a few motor parameters related with the control system, like: stator resistance, motor poles and maximum currents. They can be easily modified through a header file while no additional changes are necessary for the rest of the code. This design allows easy adaptation of the software to other motors and control platforms.

7.2.3 AD Sampling

Motor phase currents and DC bus voltage are input through A/D converters as shown in Table 1. The conversion mode for ADCs is the single mode.

Item	Conversion Ratio (Actual Value/A/D Input Value)	A/D Port
Bus voltage	0 to 500 V / 0 to 5 V	AN1
v- and w-phase currents	30 to -30 A / 0 to 5 V	AN3, AN4

Table1 AD conversions of bus voltage and motor phase currents

Basically, the signals can be categorized into two main types: bipolar and unipolar. The motor phase currents are examples of bipolar signals while the DC bus voltage is an example of unipolar signal.

The A/D inputs accept the analog input signals (i_v, i_w) in the range of 0-5 volts for SH7086 MCU with the ground referenced to 0 volt. The A/D input value of 2.5 V is taken as the current of zero, and A/D input values from 2.5V to 5V correspond to positive current values and from 0 to 2.5V negative current values.

The current is formatted in the 2^{10} . The current values could be computed by:

$$i_v = KADI * (AN3_i_v - i_{v_offset})$$

$$i_w = KADI * (AN4_i_w - i_{w_offset})$$

$$i_u = -(i_v + i_w)$$

Where i_u, i_v and i_w are three phase motor currents;

i_{v_offset} and i_{w_offset} are v and w phase current offsets;

$AN3_i_v$ and $AN4_i_w$ are ADC reading values of v and w phase currents;

$KADI$ is motor phase current scaling.

The bus voltage is in the 2^6 format and used to calculate PWM duty ratio.

$$V_{bus} = KADV * AN1_V_{bus}$$

Where V_{bus} is DC bus voltage;

$AN1_V_{bus}$ is ADC reading value of DC bus voltage;

$KADV$ is DC bus voltage scaling.

The functions in this module have:

- Current_Offset(): calculates current sensor offsets.
- Lem_Current_Calc(): calculates phase currents from current sensor readings.

7.2.4 Speed Setting and Ramp Generation

The reference speed is set through the potentiometer from minimum speed to maximum speed. The ramp generation receives the speed and direction commands from the potentiometer and applies the proper acceleration or deceleration rates to achieve the target speed. It runs in the background of the main code. As the command speed is increased or decreased, the speed regulator controls the torque production, and the torque current command is adjusted until the command speed reaches the target speed.

The functions of the module include:

- Speed_Pot() inputs the reference speed.
- Speed_Ramp() calculates the command speed.

7.2.5 Startup Procedure

Since the SVC method is based on back EMF estimation, a minimum speed is necessary to obtain the estimated back EMF value. A motor start-up subroutine, open loop startup, is developed for this purpose. When the motor is at standstill, the control system generates three phase sinusoidal voltages to start the motor. The motor spins at a fixed acceleration rate, and the vector control algorithm controls both the i_d current and the i_q current. The rotor position angle θ is incremented according to the acceleration rate. At a given time, the control switches over to the closed speed loop.

Shown in Figure 7 is motor startup block diagram, where the motor phase angle ramps up at a constant acceleration. Vector control has executed and controlled the torque component i_q current and the flux component i_d current. The desired torque required to start up the motor can be set through the header file of “customize.h”. This start up subroutine provides the constant torque to start the motor. At the end of the startup ramp, the software automatically switches over to the closed speed loop of sensorless control, taking θ from the position estimation. The startup procedure is located in the PWM interrupt.

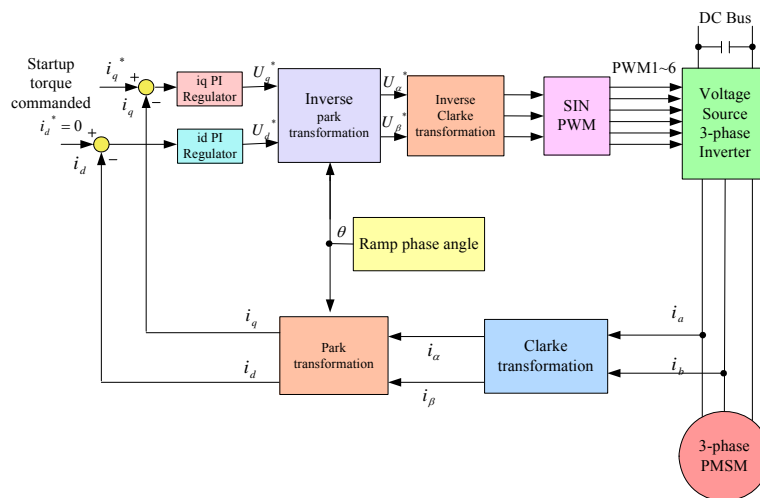


Figure 7 Block diagram of open loop startup

7.2.6 Speed and Current PI Regulator

Both speed and current regulators execute at the sampling frequency of 20kHz, and employ a traditional proportional and integral (PI) control to realize motor speed and current regulation.

The functions in the module include:

- SVC_Speed_PI(): is speed PI regulator.
- SVC_IdIq_PI(): is d and q current PI regulators.

7.2.7 PWM generation

PWM signals are generated through MTU2S as shown in Table 2.

Peripherals Used	SH 7086 (RSK + Shakti-II Pin out)		
	Signal Name	SH7086 Pin-Numbers	Port
MTU2S / TIOC3BS	M1_Up	81	PD28
MTU2S/ TIOC3DS	M1_Un	82	PD29
MTU2S/ TIOC4AS	M1_Vp	12	PE18
MTU2S/ TIOC4CS	M1_Vn	14	PE20
MTU2S / TIOC4BS	M1_Wp	13	PE19
MTU2S/ TIOC4DS	M1_Wn	15	PE21

Table2 MTU2S definition for PWM

The SH7086 port registers are defined in “hwsetup.c” as:

- PFC.PDIORH.WORD=0xB000;
- PFC.PDIORL.WORD=0x0000;
- PFC.PDCRH4.WORD=0x3033;
- PFC.PEIORH.WORD=0x003C;
- PFC.PEIORL.WORD=0x00C8;
- PFC.PECRH2.WORD=0x0011;
- PFC.PECRH1.WORD=0x1100.

The function of this module has:

- SVC_PWM_Generation() generates three phase PWM signals.

7.2.8 Protections

The purpose of this module is to protect hardware power board and motor, including hardware POE - over current protection; and software protection: over motor phase current, over DC bus voltage, and over speed. Other protections will be added later including watchdog, locked rotor, open winding, and heat sink over temperature, etc.

The functions in the module include:

- SVC_Fault_Int_IRQ0() is hardware over current protection.
- Software_Protection() is software protections for over phase current, over bus voltage, and over speed.
- SVC_PWM_Stop_Out() disables PWM outputs.

7.3 Modules in SVC Library

7.3.1 Clarke, Inverse Clarke, Park and Inverse Park Transformations

Clarke, Park and inverse Clarke and inverse Park transformations are implemented based on general vector control theory described in section 11.

The functions of this module include:

- SVC_Lib_Clarke() is Clarke transformation.
- SVC_Lib_Inverse_Clarke() is inverse Clarke transformation.
- SVC_Lib_Park() is Park transformation.
- SVC_Lib_Inverse_Park() is inverse Park transformation.

7.3.2 PWM Duty Calculation

The purpose of this module is to calculate PWM duty ratios according to motor phase voltage commands and DC bus voltage.

$$D_u = v_u / V_{bus}$$

$$D_v = v_v / V_{bus}$$

$$D_w = v_w / V_{bus}$$

Where D_u , D_v and D_w are three phase inverter duty ratios;

v_u , v_v and v_w are motor phase voltages;

V_{bus} are DC bus voltage.

The function in this module has:

- SVC_Lib_Duty_Calc () calculates PWM duty ratios.

7.3.3 Flux and Speed Estimation

As described in section 13.1, the rotor position and speed are estimated by the flux observer. In the stationary α and β reference frame, the magnetic flux is calculated from the applied voltages and the measured currents through an integration. In practice, the implementation of this procedure is problematic because of ramp-drift and dc-offset produced on the output signal. A dc-component in measured motor back EMF is inevitable in experiments. This dc-component, no matter how small it is, can finally drive the pure integrator to saturation. Furthermore, the initial value problem associated with the pure integrator occurs from the integral value at the initial instant.

To solve these issues and avoid complexity at the same time, a low pass filter is used. When the dc-component measured within the back EMF is weak, only one low pass filter works well for estimating the flux. In the opposite case, we are confronted with the problem of an offset persisting in the output $\lambda_{\alpha\beta}$. At this stage, a second low pass filter is used.

Therefore, instead of direct integration of back EMF, the flux is estimated through two low pass filters in order to remove the integration error and the DC offset according to flux equations:

- First low pass filter
- Derivative
- Second low pass filter

The complete necessary arrangement is illustrated in Figure 8.

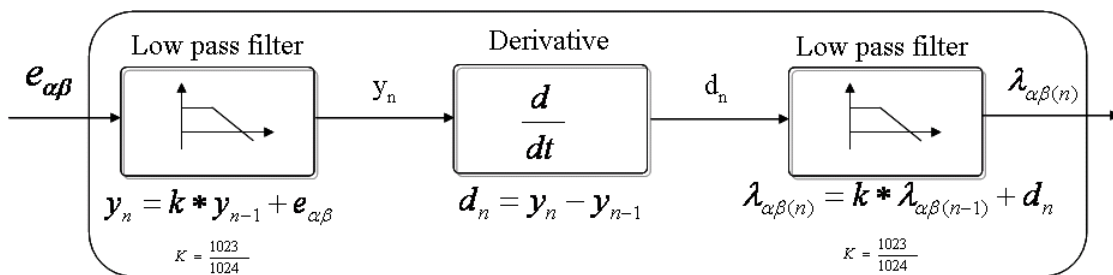


Figure 8 Flux observer implementation

Once the fluxes are estimated, the rotor angular position can be calculated from the flux components in the stationary α and β reference frame.

$$\theta = a \tan\left(\frac{\lambda_\beta}{\lambda_\alpha}\right)$$

The speed is obtained from the phase difference $\omega_r = \frac{d\theta}{dt}$, and then filtered with a third order low pass filter using three first order low pass filters.

The functions of this module include:

- SVC_Flux_Est_Par() sets up flux filter parameters.
- SVC_Lib_Flux_Est() estimates the flux by low pass filters.
- SVC_Lib_Phase_Est() calculates the rotor position.
- SVC_Speed_Est_Par () sets up speed estimation filter parameters.
- SVC_Lib_Speed_Est() calculates the speed by LPFs.

8. Software Descriptions

The SVC software has the following features:

- All codes are written in C language;
- The software is modularized according to the SVC block diagram (as shown in Figure 2);
- Core SVC modules can be generally used without any changes;
- I/O definitions and basic MCU drivers are automatically generated by HEW;
- ADC scaling, motor and control parameters are easily tuned through a header file of “customize.h”;
- Watch window allows real-time viewing of variables.

8.1 SVC Software Workspace

Shown in Figure 9 is the workspace for SVC using Renesas’ HEW. The codes include dbstc.c; hwsetup.c, intprg.c; lcd.c; main.c; motorcontrol.c; resetprg.c and vectbl.c. Vector control transformations and speed and position observer are put in the library of SH_SVC_Lib_001.lib.

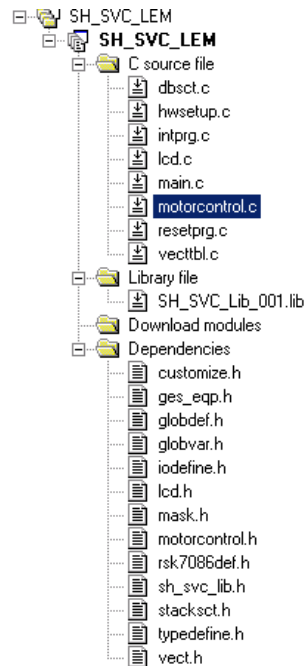


Figure 9 SVC software workspace

- dbstc.c includes structures used by the runtime library both to clear un-initialized global variables and to write initial values into initialized global variable sections.
- hwsetup.c is hardware initializations.
- vecttbl.c contains the array of addresses of ISRs.
- resetpr.c has functions called just after reset.
- intprg.c is entry points for all of standard ISRs vectors.

All functions above can be automatically generated by HEW.

- lcd.c includes LCD display functions.
- main.c, motorcontrol.c and SH_SVC_Lib_001.lib are described in detail in the following sections.

8.2 Main Code

Shown in Figure 10 are function calls in main.c including: initialization of SVC parameters; LCD display; MTU2S PWM registers; and flux and speed estimation parameters. The current sensor offsets are calculated before the output of PWMs. In the while loop, actual and reference speeds are displayed every 250μS, and the PWM interrupt of MTU2S_PWM_ISR is executed every 50μS.

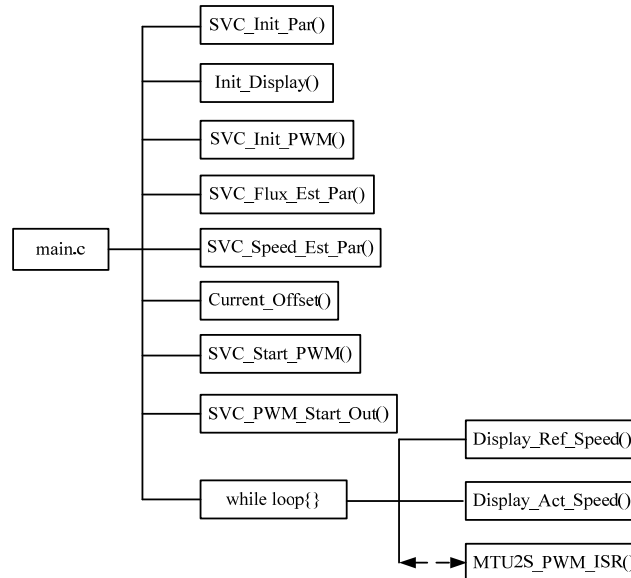


Figure 10 Block diagram of function calls in main.c

8.3 Motor Control Code

The functions in motorcontrol.c are plotted in Figure 11. The motorcontrol.c is a major code for SVC, which contains all functions and function calls to implement SVC.

- Current_Offset() calculates current sensor offsets.
- Lem_Current_Calc measures motor phase currents.
- Delay() generates the time delay.
- Wait_Sync() is a time synchronization function.
- SVC_Init_PWM() initializes MTU2S PWM registers.
- SVC_Start_PWM() starts PWMs and enables the PWM interrupt.
- SVC_PWM_Start_Out() enables PWM outputs.
- SVC_Init_Par() initializes control parameters.
- SVC_Var_Clamp is to clamp individual variable values between two limits.
- SVC_Speed_PI() is speed PI regulator.
- SVC_IdIq_PI() is current Id and Iq regulators.
- SVC_PWM_Generation() generates 6 PWM signals.
- Speed_Ramp() accelerates and decelerates the reference speed to the desired speed.
- The PWM interrupt of MTU2S_PWM_ISR() executes the SVC algorithm – startup, speed loop, current loops, vector control transformation and PWM generation.
- SVC_Fault_Int_IRQ0() is a interrupt of hardware over current POE protection.
- ADC_Measurement() samples motor phase currents and DC bus voltage.
- Soft_Protection() is software protections regarding over current, over bus voltage, over speed, etc.
- Speed_pot() gives a reference speed through a potentiometer.
- Display_Act_Speed() and Display_Ref_Speed display the actual and reference speeds, respectively.

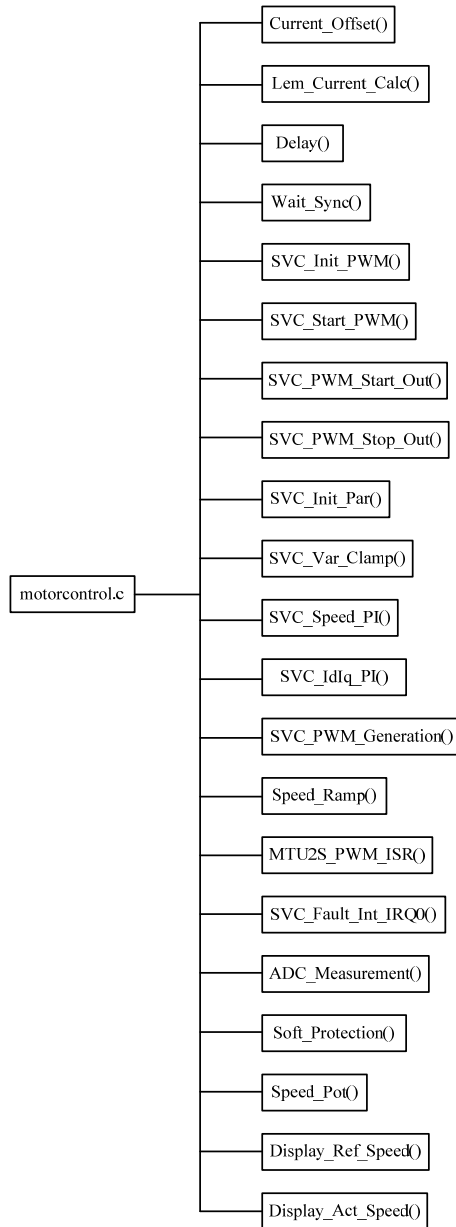


Figure 11 Functions in motorcontrol.c

8.4 SVC Library Code

Functions in the SVC library of SH_SVC_Lib_001.lib include PI controller of SVC_Lib_PI(); PWM duty calculation of SVC_Lib_Duty_Calc(); vector control transformations: SVC_Lib_Clarke(), SVC_Lib_Inverse_Clarke(), SVC_Lib_Park(), and SVC_Lib_Inverse_Park(); phase and speed estimation resets of SVC_Lib_phase_reset() and SVC_Lib_speed_reset(); as well as flux, position and speed estimation functions of SVC_Lib_Flux_Est(), SVC_Lib_Phase_Est(), and SVC_Lib_Speed_Est(). The SVC_Lib_angle_set() transfers the phase angle to the sine and cosine values for Park and inverse Park transformations.

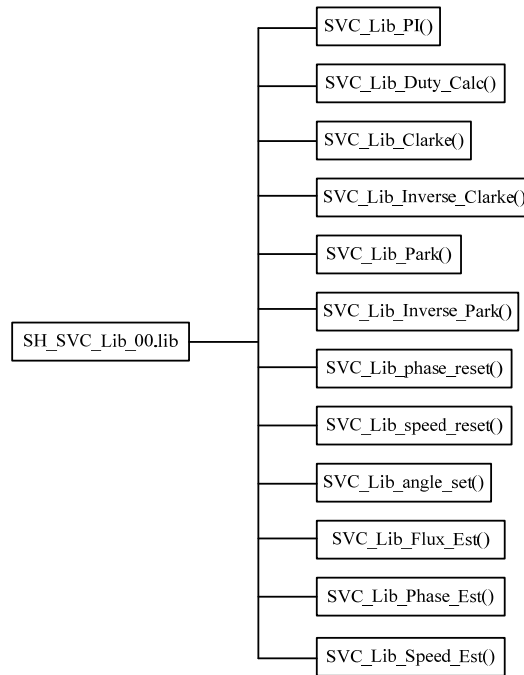


Figure 12 Functions in SVC library

8.5 Flowchart of PWM Interrupt

MTU2S_PWM_ISR() is a major function to implement SVC. Figure 13 is a flowchart of PWM interrupt. It starts with the open loop, and then switches to the closed speed loop.

The procedures in the PWM interrupt of MTU2S_PWM_ISR() are:

- Motor phase motor currents and DC bus voltage are first sampled;
- When the motor powers on, the startup procedure handles the open loop starting;
- After the motor starts up at the given time, the system switched over to the closed speed loop;
- The rotor position and speed are estimated in sync with the carrier frequency in order to update the position and the speed timely;
- The current PI controller outputs of v_d and v_q are transformed back to three phase voltages of v_u , v_v and v_w , which are used to calculate PWM duty ratios to drive motor to the desired voltage.

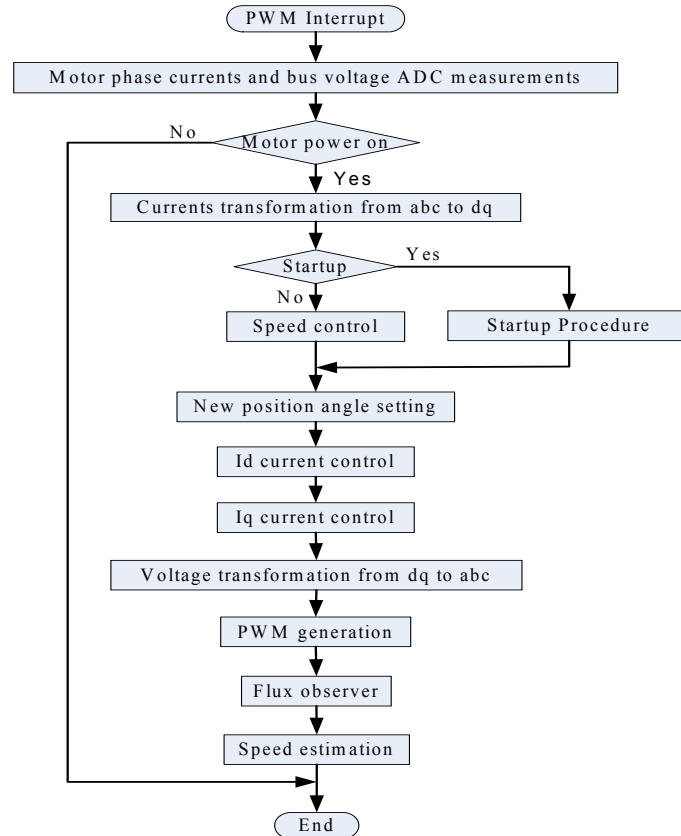


Figure 13 Flowchart of PWM interrupt

9. Motor and Control Parameters

9.1 Parameters in customize.h

This section describes how to modify motor and control parameters for different motors and applications. All parameters that need to be changed can be modified via the header file of “customize.h”.

PWM_FREQ_CUSTOM

Purpose: This macro defines PWM frequency (Hz).

Valid Values: 500 to 40000

Default: 20000

Example: #define PWM_FREQ_CUSTOM 10000

SAMPLE_FREQ_CUSTOM

Purpose: This macro defines sample frequency (Hz).

Valid Values: 500 to 40000

Default: 20000

Example: #define SAMPLE_FREQ_CUSTOM 20000

SWITCH_ACTIVE

Purpose: This macro defines power switch devices to be active low (0) or active high (1).

Valid Values: 0 or 1

Default: 0

Example: #define SWITCH_ACTIVE 0

DEAD_TIME_CUSTOM

Purpose: This macro defines the dead time of power switch devices to avoid the short circuit between the top and bottom power switches (μS). Note: This value is a function of hardware inverter stage.

Valid Values: 0.0 to 10.0

Default: 1.0

Example: #define DEAD_TIME_CUSTOM 1.5

POLES_CUSTOM

Purpose: This macro defines pair of poles of motor.

Valid Values: 1 to 100

Default: 2

Example: #define POLES_CUSTOM 2

R_STA_CUSTOM

Purpose: This macro defines stator resistance of motor. The value is multiple of 10 (Ω).

Valid Values: 1 to 100

Default: 50

Example: #define R_STA_CUSTOM 50

I_START_CUSTOM

Purpose: This macro defines maximum startup current (amps) of motor. The value is multiple of 10.

Valid Values: 0.1 to 100

Default: 1

Example: #define I_START_CUSTOM 10

IQ_MAX_CUSTOM

Purpose: This macro defines maximum torque current at normal operation (amps). The value is multiple of 10.

Valid Values: 0. to 100

Default: 5

Example: #define IQ_MAX_CUSTOM 5

RPM_MIN_CUSTOM

Purpose: This macro defines minimum speed of motor (rpm).

Valid Values: 100 to 10000

Default: 500

Example: #define RPM_MIN_CUSTOM 500

RPM_MAX_CUSTOM

Purpose: This macro defines maximum speed of motor (rpm).

Valid Values: 1,000 to 30000

Default: 3500

Example: #define RPM_MAX_CUSTOM 3500

KADI

Purpose: This macro scales motor phase currents, which depends on hardware platform.

Format: 2^{10} , 1amps to 1024

Renesas' Power Board: 15360 for LEM current sensor

Example: #define KADI 15360

KADV

Purpose: This macro scales DC bus voltage, which depends on hardware platform too.

Format: 2^6 , 1volt to 64

Renesas' Power Board: 15658

Example: #define KADV 15658

STARTUP_RAMPTIME_CUSTOM

Purpose: This macro defines motor startup ramp time (mS).

Valid Values: 1000 to 5000

Default: 1000

Example: #define STARTUP_RAMPTIME_CUSTOM 1000

R_ACC_CUSTOM

Purpose: This macro defines motor startup acceleration rate in rpm/sec.

Valid Values: 1 to 10000

Default: 1000

Example: #define R_ACC_CUSTOM 1000

KP_CUR_CUSTOM

Purpose: This macro defines the proportional gain of the current regulator.

Valid Values: 10 to 10000

Default: 200

Example: #define KP_CUR_CUSTOM 200

KI_CUR_CUSTOM

Purpose: This macro defines the integral gain of the current regulator.

Valid Values: 10 to 10000

Default: 150

Example: #define KI_CUR_CUSTOM 150

KP_SPD_CUSTOM

Purpose: This macro defines the proportional gain of the speed regulator.

Valid Values: 10 to 10000

Default: 200

Example: #define KP_SPD_CUSTOM 200

KI_SPD_CUSTOM

Purpose: This macro defines the integral gain of the speed regulator.

Valid Values: 10 to 10000

Default: 150

Example: #define KI_SPD_CUSTOM 150

FIRST_FLUX_LOWPASS_TIME_CUSTOM

Purpose: This macro defines the time constant of the first flux low pass filter, which is $T_c * 2^{\text{LOWPASS_TIME_CUSTOM}}$, where T_c is the sampling time.

Valid Values: 1 to 100

Default: 10

Example: #define FIRST_FLUX_LOWPASS_TIME_CUSTOM 10

LAST_FLUX_LOWPASS_TIME_CUSTOM

Purpose: This macro defines the time constant of the second flux low pass filter.

Valid Values: 1 to 100

Default: 10

Example: #define LAST_FLUX_LOWPASS_TIME_CUSTOM 10

DERIVATIVE_TIME_CUSTOM

Purpose: This macro defines the time constant of the derivative, which is realized as the difference between the current n^{th} sample and the previous $(n - \text{DERIVATIVE_TIME_CUSTOM})^{\text{th}}$ sample.

Valid Values: 1 to 10

Default: 1

Example: #define FIRST_FLUX_LOWPASS_TIME_CUSTOM 1

FIRST_SPEED_LOWPASS_TIME_CUSTOM

Purpose: This macro defines the time constant of the first speed low pass filter, which is

$T_c * 2^{\text{FIRST_SPEED_LOWPASS_TIME_CUSTOM}}$, where T_c is the sampling time. The speed estimation is obtained from the phase difference, and then filtered with a third order low pass filter using three first order low pass filters.

Valid Values: 1 to 100

Default: 5

Example: #define FIRST_SPEED_LOWPASS_TIME_CUSTOM 5

SECOND_SPEED_LOWPASS_TIME_CUSTOM

Purpose: This macro defines the time constant of the second speed low pass filter, which is

$$T_c * 2^{\text{SECOND_SPEED_LOWPASS_TIME_CUSTOM}}, \text{ where } T_c \text{ is the sampling time.}$$

Valid Values: 1 to 100

Default: 4

Example: #define SECOND_SPEED_LOWPASS_TIME_CUSTOM 4

THIRD_SPEED_LOWPASS_TIME_CUSTOM

Purpose: This macro defines the time constant of the third speed low pass filter, which is

$$T_c * 2^{\text{THIRD_SPEED_LOWPASS_TIME_CUSTOM}}, \text{ where } T_c \text{ is the sampling time.}$$

Valid Values: 1 to 100

Default: 3

Example: #define THIRD_SPEED_LOWPASS_TIME_CUSTOM 3

9.2 Operation Variables

In order to run SVC, the following variables should be initially set up or monitored through the user interface or the real time watch window:

start_flg

Purpose: It is used to start up SVC. The value of 0 stops SVC. The value of 1 is to start up SVC.

Valid Values: 0 or 1

Default: 0

fault_flg

Purpose: It shows system faults. The value 1 represents any faults. The value 0 indicates no faults.

Valid Values: 0 or 1

Default: 0

ref_speed

Purpose: It sets up motor speed (rpm), which can be input through the user interface, the potentiometer, or the real time watch window.

Valid Values: 500 or 30000

Default: 500

act_speed

Purpose: It is the motor actual speed (rpm), which is estimated from motor phase currents using the SVC algorithm.

Valid Values: 0 or 30000

Default: 0

Refer to the source code for other operation variables, such as: DC bus voltage of vbus, motor currents of iu, iv, iw, id, and iq; the rotor position of Phase_est, and the estimated speed of Speed_est, etc.

9.3 Parameter Tuning Example

Shown in Figure 14 are a BLDC motor and its data sheet. The motor is a 2-pole 3-phase BLDC motor. The rated power is 0.25 HP. The maximum speed is 2500 rpm. According to the data sheet, motor and control parameters have to be properly modified to run SVC.



Figure 14 A physical BLDC motor for parameter tuning

Motor Pole	4
Phase	3
Voltage	130 V
Current	1.5 A
Power	1/5 hp
Speed	2500 rpm
Inductance	27 mh
Stator Resistor	5.1Ω
Hall sensors	3

Table3 Motor data sheet

First, define motor parameters:

- #define R_STA_CUSTOM 51 // stator phase resistance 5.1Ω /10
- #define POLES_CUSTOM 2 // 2 pair of poles
- #define I_START_CUSTOM 15 // startup current of 1.5A in Amps/10
- #define IQ_MAX_CUSTOM 45 // maximum iq current of 4.5A in Amps/10
- #define RPM_MIN_CUSTOM 500 // minimum motor speed of 500rpm
- #define RPM_MAX_CUSTOM 2500 // maximum motor speed of 2,500rpm

Second, modify control parameters related with hardware platform:

- #define PWM_FREQ_CUSTOM 20000 // PWM Frequency in 20,000Hz
- #define SAMPLE_FREQ_CUSTOM 20000 // Sample Frequency in 20,000 Hz
- #define DEAD_TIME_CUSTOM 2.0 //Switch dead time is 2. μS.
- #define SWITCH_ACTIVE 0 //Switch is the active low.
- #define KADI 15360 //Current sensor scaling
- #define KADV 15658 //DC bus voltage scaling

Last, tune control parameters:

- #define R_ACC_CUSTOM 1000 // acceleration ramp in 1000rpm/sec
- #define KP_CUR_CUSTOM 200 // proportional gain of current controller
- #define KI_CUR_CUSTOM 100 // integral gain of current controller
- #define KP_SPD_CUSTOM 300 // proportional gain of speed controller
- #define KI_SPD_CUSTOM 200 // integral gain of speed controller
- #define FIRST_FLUX_LOWPASS_TIME_CUSTOM 10
- #define DERIVATIVE_TIME_CUSTOM 1
- #define LAST_FLUX_LOWPASS_TIME_CUSTOM 10
- #define FIRST_SPEED_LOWPASS_TIME_CUSTOM 5

- #define SECOND_SPEED_LOWPASS_TIME_CUSTOM 4
- #define THIRD_SPEED_LOWPASS_TIME_CUSTOM 3

10. MCU Performance Analysis

SH7086 MCU performance is demonstrated in the following chart. Shown in Figure 15 is the CPU bandwidth with the PWM frequency. The blue bar represents the CPU usage of SVC, while the purple bar represents the free CPU bandwidth. At a PWM frequency of 20KHz, the CPU bandwidth is about 75%, with 25% left for adding other functions. If the PWM frequency is reduced to 10KHz, the CPU bandwidth drops to 35%.

SH7086 MCU has an interrupt skipping function for both MTU2 and MTU2S. With this feature, Channel 3 compare-match interrupt and Channel-4 underflow interrupt could be skipped several times in the complementary PWM mode. The interrupt skipping value is defined in the header file of “customize.h”. If PWM_FRE_CUSTOM is not the same as SAMPLE_FRE_CUSTOM, the INT_SKIP equals the ratio of these two variables. For instance, if the PWM frequency is set at 20KHz and the control loop is run with 10KHz, the PWM interrupt is skipped twice.

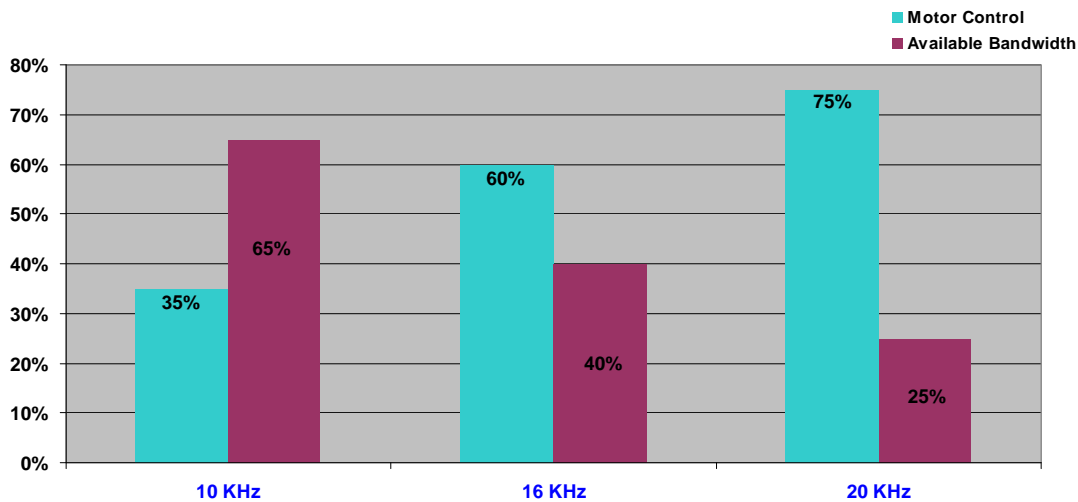


Figure 15 CPU performance analysis of SVC

11. Vector Control Theory

VC controls stator currents represented by a vector. It is based on projections which transform a three-phase time and speed dependent system into a two axis (d and q axis) time invariant system. These projections lead to a structure of AC motor drives similar to that of a DC motors. VC has two constants as input references: the torque component, which is aligned with the q axis; and the flux component, which is aligned with d axis. Because VC is simply based on projections, the control structure handles instantaneous electrical quantities. This makes VC accurate at the dynamic and steady and independent of the limited bandwidth mathematical model.

Therefore, VC controls motors in the following way:

- The flux is controlled by the d axis current: $\lambda = k_{\lambda} i_d$;
- The torque is controlled by the q-axis current: $T = k_T \lambda i_q$.

By maintaining the amplitude of the flux at a fixed value there is a linear relationship between the torque and the torque component current. The torque could be directly controlled by the torque component of the stator current vector.

11.1 Space vector definition

Three-phase voltages and currents of PMSM motors can be analyzed in terms of complex space vectors. With regard to three-phase stator currents of i_u , i_v , and i_w , the space vector can be defined as follows. Assuming that i_u , i_v , and i_w are instantaneous currents in stator phases, then the complex stator current vector I_s is defined by:

$$I_s = i_u + \alpha i_v + \alpha^2 i_w$$

Where $\alpha = e^{j\frac{2\pi}{3}}$ represents the spatial operator. The following diagram shows the stator current complex space vector:

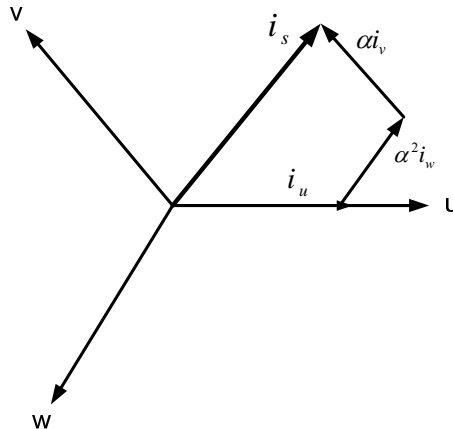


Figure 16 Space current vector definition

Where u , v , and w are three phase system axes. This space vector depicts the three-phase sinusoidal system. It needs to be transformed into a two time invariant axis system. The transformation can be split into two steps:

- Clarke transformation from u , v , and w to α and β , which outputs a two-axis time variant system.
- Park transformation from α and β to d and q , which outputs a two co-ordinate time invariant system.

11.2 Clarke Transformation

The space vector can be projected in the stationary reference frame with only two orthogonal axis called α and β system. Assuming that the u axis and the α axis are in the same direction, the following Figure shows the vector diagram. The projection that modifies the three-phase system of u , v , and w into the α and β two dimension orthogonal system is presented below.

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_u \\ i_v \\ i_w \end{bmatrix}$$

Where $i_u + i_v + i_w = 0$, thus the equations can be expressed as:

$$\begin{cases} i_\alpha = i_u \\ i_\beta = \frac{\sqrt{3}}{3} i_u + \frac{2\sqrt{3}}{3} i_v \end{cases}$$

In the two dimension stationary frame, the α and β currents are still time variant variables and depend on the time and the speed.

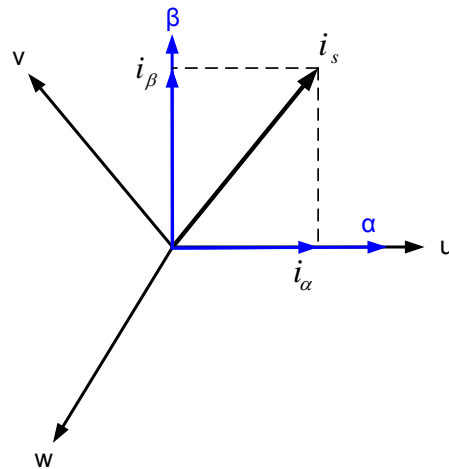


Figure 17 Clarke transformation

11.3 Park Transformation

Another transformation is the Park transformation in VC. In fact, this projection modifies a two phase orthogonal α and β system into the d and q rotating reference frame. If the d -axis is considered aligned with the rotor flux, the following diagram shows, for the current vector, the relationship between the α and β reference frame and the d and q reference frame:

where θ is the angle between the d -axis and the α -axis. The d and q components of the current vector are determined by the following equations:

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

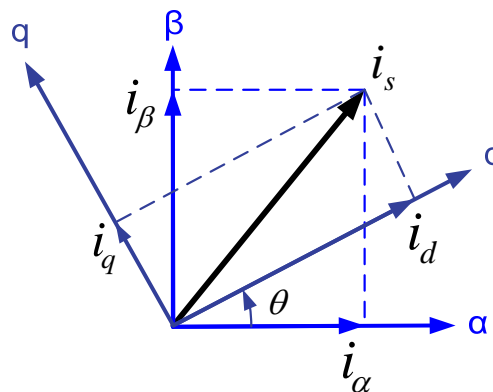


Figure 18 Park transformation

These components depend on the current vector α and β components and the rotor position. If the right rotor position is detected by this projection, the d and q components become a constant. Thus the d and q two co-ordinate system has the following characteristics:

- two coordinate time invariant system;
- with the i_d current and the i_q current, the flux and the torque could be directly controlled.

11.4 Inverse Park Transformation

VC has two voltage components in the rotating d and q reference frame, which need to go through complementary inverse transformations to get back to three phase motor terminal voltages. It is transformed from the two axis rotating d and q frame to the two-axis stationary α and β frame. This transformation is called the inverse Park transformation.

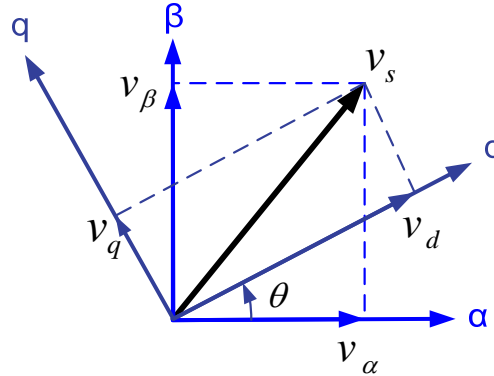


Figure 19 Inverse park transformation

The voltage transformation equations are given as:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} v_d \\ v_q \end{bmatrix}$$

11.5 Inverse Clarke Transformation

The inverse Clarke transformation is to transform from the two-axis α and β stationary frame to the three-phase reference three-axis u , v , and w stationary frame of the stator to get the three phase motor terminal voltages of v_u , v_v , and v_w . This transformation is mathematically illustrated in Figure 20.

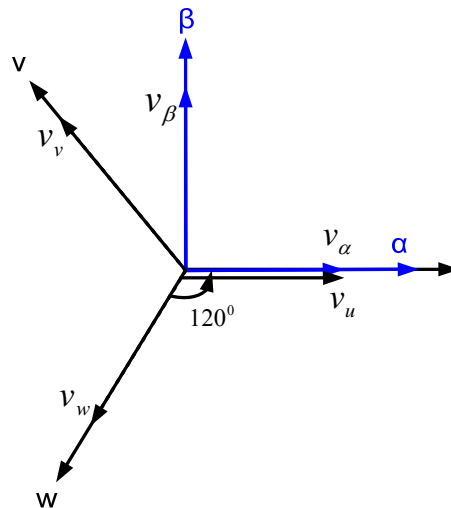


Figure 20 Inverse Clarke transformation

The transformation equation is described as:

$$\begin{bmatrix} v_u \\ v_v \\ v_w \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix}$$

12. Modeling Vector Control of PMSM Motors

Three-phase PMSM motors can be modeled in various reference frames such as the three-phase u , v , and w stationary frame or the rotating d and q reference frame. In the three-phase u , v , and w stationary frame, the motor equations obtained are nonlinear and strongly coupled with each other, making it very difficult to perform direct control of the flux and the torque by the motor currents. In this application, the motor is modeled in the rotating d and q reference frame so that it is capable of decoupling three phase stator currents into the flux component and the torque component, and then directly controlling the flux and the torque like a separately excited DC motor. It enables the performance of PMSM motor drives to be comparable or even superior to that of DC motor drives.

The mathematical model of the PMSM motor in the synchronous rotating reference frame aligned with the rotor flux linkage can be expressed as follows:

$$v_d = r_s i_d - \omega_r \lambda_q + p \lambda_d$$

$$v_q = r_s i_q + \omega_r \lambda_d + p \lambda_q$$

$$\lambda_d = L_s i_d + \lambda_m$$

$$\lambda_q = L_s i_q$$

$$T_e = \frac{3P}{2} (\lambda_d i_q - \lambda_q i_d)$$

Where v_d and v_q are stator voltages in the d - q axes;

i_d and i_q are stator currents in the d - q axes;

λ_d and λ_q are stator flux linkages in the d - q axes;

λ_m is permanent magnet flux linkage;

T_e is electromagnetic torque;

ω_r is angular velocity of rotor;

r_s is stator resistance;

L_s is stator self inductance;

P is number of poles;

$$p = \frac{d}{dt}.$$

Since the rotor flux is oriented with the d -axis and the i_d current of the PMSM motor should be controlled to be zero, the stator flux linkage and the torque equations become:

$$\lambda_d = \lambda_m$$

$$T_e = \frac{3P}{2} \lambda_m i_q$$

Therefore, the VC of the three-phase PMSM motor enables the flux λ_d and the torque T_e to be decoupled, respectively, and thus independent control of the flux and the torque can be achieved.

13. Sensorless Vector Control Methodology

For Renesas' SVC implementation, the sensorless technique adopts an open-loop voltage model. Based on the motor model, it estimates the flux and position by integrating motor back EMF. Instead of direct integration of back EMF, the flux is estimated through two low pass filters in order to remove integration error and DC offset. The advantages of this implementation are:

- parameter insensitive, only stator resistance;
- eliminating integration error and DC offset;
- no position estimation offset;
- no initial position and flux requirements;
- simple to implement and less computation.

13.1 Flux and Position Estimation

The position and the speed are estimated based on the motor model in the stationary α and β reference frame. The voltage and flux equations of PMSM motors are expressed as:

$$v_{\alpha} = r_s i_{\alpha} + p \lambda_{\alpha}$$

$$v_{\beta} = r_s i_{\beta} + p \lambda_{\beta}$$

The flux can be calculated from applied voltages and measured currents by integration as:

$$\lambda_{\alpha} = \lambda_{\alpha 0} + \int_0^t (v_{\alpha} - r_s i_{\alpha}) dt$$

$$\lambda_{\beta} = \lambda_{\beta 0} + \int_0^t (v_{\beta} - r_s i_{\beta}) dt$$

Also according to the motor equivalent circuit in the stationary α and β reference frame, the flux can be described as;

$$\lambda_{\alpha} = L_s i_{\alpha} + \lambda_{am} = L_s i_{\alpha} + \lambda_m \cos(\theta_r)$$

$$\lambda_{\beta} = L_s i_{\beta} + \lambda_{\beta m} = L_s i_{\beta} + \lambda_m \sin(\theta_r)$$

Where v_{α} and v_{β} are stator voltages in the $\alpha - \beta$ axes;

i_{α} and i_{β} are stator currents in the $\alpha - \beta$ axes;

λ_{α} and λ_{β} are stator flux linkages in the $\alpha - \beta$ axes;

θ_r is rotor angle that equals $\omega_r t$.

Furthermore, the flux equations can be written as:

$$\lambda_m \cos(\theta_r) = \lambda_{\alpha} - L_s i_{\alpha}$$

$$\lambda_m \sin(\theta_r) = \lambda_{\beta} - L_s i_{\beta}$$

Since the phase inductance L_s is normally small, the inductance contribution in the above equations can be neglected.

The flux equations can be simplified:

$$\lambda_m \cos(\theta_r) = \lambda_{\alpha}$$

$$\lambda_m \sin(\theta_r) = \lambda_{\beta}$$

Therefore, the rotor angular position can be calculated from the flux components in the stationary α and β reference frame.

$$\theta_r = \arctg^{-1} \left(\frac{\lambda_{\beta}}{\lambda_{\alpha}} \right)$$

13.2 Speed Estimation

The motor speed ω_r is deduced from the derivative of the angle θ_r .

$$\omega_r = \frac{d}{dt} \theta_r$$

Based on the equations above, the sensorless algorithm is developed by the applied phase voltages, the measured phase currents to estimate the rotor position θ_r and the speed ω_r .

14. References

- Renesas Technology Website: <http://www.renesas.com/>
- Renesas' Application Note: MCRP05: Brushless AC Motor Control Reference Platform (REG05B0051-0100).
- Renesas' DevCon presentation of "Sensorless Vector Control of Permanent Magnet Synchronous Motors", DevCon Conference, Oct. 2008, San Diego, California, USA.
- Renesas' white paper of "Understanding Sensorless Vector Control for Brushless DC Motors", ESC-2008, Embedded System Silicon Valley conference, April 15-17, 2008, San Jose, California, USA.
- Renesas' white paper of "Sensorless Vector Control of PMSM Motor using One Shunt Current Detection", IEEE-IAS 2008, Oct. 5-9, Edmonton, Alberta, Canada.
- Renesas' white paper of "MCU Performance for Various Control Algorithms of BLDC Motors", eDrive conference, Feb. 14-15, 2008, Atlanta, Georgia, USA.

Website and Support

Renesas Technology Website
<http://www.renesas.com/>

Inquiries
<http://www.renesas.com/inquiry>
csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Mar. 06,2009	32	Initial version

All trademarks and registered trademarks are the property of their respective owners.

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (<http://www.renesas.com>)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
 - (1) artificial life support devices or systems
 - (2) surgical implantations
 - (3) healthcare intervention (e.g., excision, administration of medication, etc.)
 - (4) any other purposes that pose a direct threat to human life
 Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

© 2009. Renesas Technology Corp., All rights reserved.