RX Family

Vector Control for Permanent Magnet Synchronous Motor with Encoder - For MCK

Summary

This application note is intended to describe a sample program that uses Renesas RX26T microcontroller to drive a permanent magnet synchronous motor with an encoder under vector control. The software targeted for this application note uses a smart configurator, and in particular uses a motor component that provides a driver for the multifunction timer pulse unit and 12-bit A/D converter required for motor control.

The target software of this application note is for reference only, and we do not guarantee its operation. When using the software subject to this application note, please use it after sufficient evaluation in an appropriate environment.

Operation check device

We have confirmed the operation of the target software of this application note on the following device.

- MCU used:
- RX26T RAM64KB Version(R5F526TFCDFP)
- RX26T RAM48KB Version(R5F526TACDFM)
- RX14T(R5F514T5AGFM)

Target software

The target software of this application note is shown below.

(RX26T RAM64KB Version)
- RX26T_MCBA_MCILV1_SPM_ENCD_FOC_CSP_V110 (IDE: CS+ version)
- RX26T_MCBA_MCILV1_SPM_ENCD_FOC_E2S_V100 (IDE: e^2 studio version)
- RX26T_MCBA2_MCILV1_SPM_ENCD_FOC_CSP_V100 (IDE: CS + version)
- RX26T_MCBA2_MCILV1_SPM_ENCD_FOC_E2S_V100 (IDE: e^2 studio version)

(RX26T RAM48KB Version)
- RX26T_MCBC_MCILV1_SPM_ENCD_FOC_CSP_V110 (IDE: CS+ version)
- RX26T_MCBC_MCILV1_SPM_ENCD_FOC_E2S_V110 (IDE: e^2 studio version)

Encoder vector control software for Renesas Flexible Motor Control Kit & RX26T CPU board

- RX14T_MCILV1_SPM_ENCD_FOC_CSP_V100 (IDE: CS+ version)
- RX14T_MCILV1_SPM_ENCD_FOC_E2S_V100 (IDE: e2 studio version)

Encoder vector control software for Renesas Flexible Motor Control Kit & RX14T CPU board

Contents

1. Overview 4

2. Development environment 5
2.1 Operation check environment 5
2.2 Hardware specifications 6

3. Quick Start Guide 12
3.1 Downloading and writing sample programs 12
3.2 Start Analyzer and RMT file 12
3.3 List of variables for Analyzer function 14
3.4 RMW UI operation 15
3.5 Board UI operation 20

4. Software 21
4.1 Software specification 21
4.2 Software configuration 22
4.3 File/folder structure 25

5. Functionality 30
5.1 Application layer 30
5.2 Manager module 41
5.3 Current control module 61
5.4 Modulation (current control module) 70
5.5 Voltage error compensation (current control module) 72
5.6 Speed control module 74
5.7 Magnetic flux weakening control (speed control module) 81
5.8 Disturbance torque/speed estimation observer (speed control module) 82
5.9 Position control module 83
5.10 Position profile (position control module) 93
5.11 IPD module 97
5.12 Sensor module (encoder) 100
5.13 Driver module 113
5.14 Smart Configurator settings 118

6. Vector control algorithm 128
6.1 Analysis model of permanent magnet synchronous motor 128
6.2 dq axis model of permanent magnet synchronous motor 129
6.3 Vector control system and controller 131

7. Test results 137
7.1 Program size 137

7.2 CPU load factor 137
7.3 Operating waveform 138

8. Reference materials 140

Revision History 141

1. Overview

The purpose of this application note is to explain how to use a sample program that uses a Renesas microcontroller (MCU) to drive a permanent magnet synchronous motor with an encoder by vector control. The sample program can be combined with the Motor Solution Kit (Renesas Flexible Motor Control Kit) to control the motor. In addition, it is compatible with the motor control development support tool "Renesas Motor Workbench" which can be used to check internal data of the MCU and as a user interface (UI) for motor control. Please refer to the MCU function assignment of the sample program and the interrupt load status of the control, and use them as a reference to select the MCU to be used and software development.

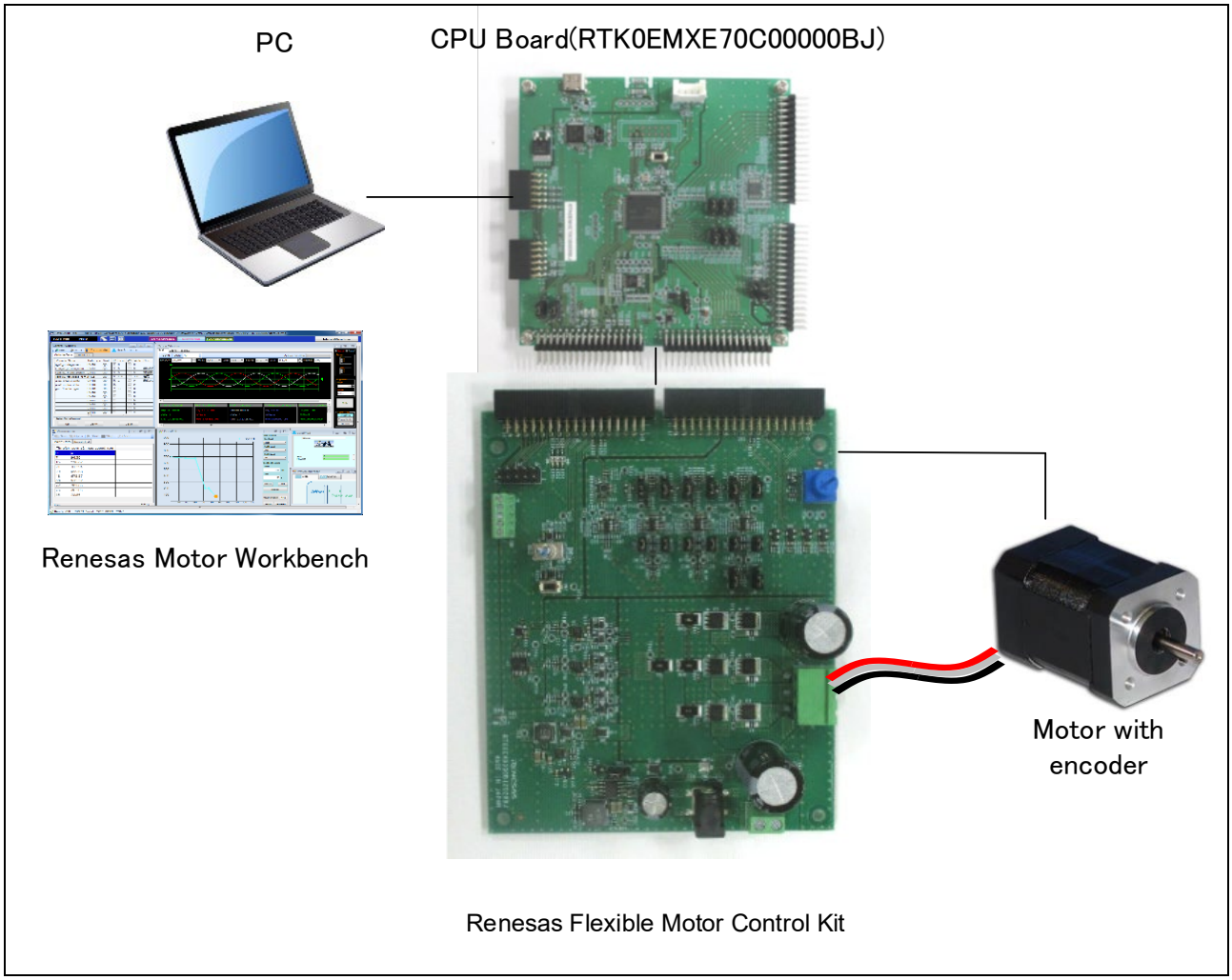


Figure 1-1 Operating environment using the sample program

2. Development environment

2.1 Operation check environment

Table 2-1,Table 2-2,Table 2-3 and Table 2-4 show the development environment of the target hardware and software.

Table 2-1 Hardware development environment (H/W)

Classification	Product used
Microcontroller / CPU board P/N	RX26T RAM64KB Version (R5F526TFCDFP) / Version 1: RTK0EMXE70C00000BJ Version 2: RTK0EMXE70C00001BJ
	RX26T RAM48KB Version (R5F526TACDFM) / Version 1: RTK0EMXE30C00000BJ Version 2: RTK0EMXE30C00001BJ
	RX14T (R5F514T5AGFM) / RTK0EMXH30C00000BJ
Inverter board	Renesas MCI-LV-1 Inverter board for 48V 10A BLDC / RTK0EM0000B12020BJ
Motor	BLY171D-24V-4000 (manufactured by Anaheim Automation)
Sensor	Encoder: AMT102-V (manufactured by CUI DEVICES)

Table 2-2 Software development environment (RX26T_MCBA_MCILV1_SPM_ENCD_FOC_CSP_V110, RX26T_MCBA_MCILV1_SPM_ENCD_FOC_E2S_V100)

IDE version	RX smart configurator	Toolchain version
CS+: V8.10.00	Version 2.18.0	CC-RX: V3.05.00
e^2 studio: 2023-07	e^2 studio plug-in version	

Table 2-3 Software development environment(RX26T_MCBA2_MCILV1_SPM_ENCD_FOC_CSP_V100, RX26T_MCBA2_MCILV1_SPM_ENCD_FOC_E2S_V100)

IDE version	RX smart configurator	Toolchain version
CS+：V8.13.00	Version 2.25.0	CC-RX：V3.07.00
e^2 studio：2025-04	e^2 studio plug-in version	

Table 2-4 Software development environment(RX14T_MCILV1_SPM_ENCD_FOC_CSP_V100, RX14T_MCILV1_SPM_ENCD_FOC_E2S_V100)

IDE version	RX smart configurator	Toolchain version
CS+：V8.14.00	Version 2.27.0	CC-RX：V3.07.00
e^2 studio：2025-10	e^2 studio plug-in version	

Table 2-5 Software development environment(RX26T_MCBC_MCILV1_SPM_ENCD_FOC_CSP_V110, RX26T_MCBC_MCILV1_SPM_ENCD_FOC_E2S_V110)

IDE version	RX smart configurator	Toolchain version
CS+：V8.15.00	Version 2.29.0	CC-RX：V3.07.00
e^2 studio：2026-04.3	e^2 studio plug-in version	

For purchase and technical support, please contact our Sales and Distributors.

2.2 Hardware specifications

2.2.1 Hardware configuration diagram

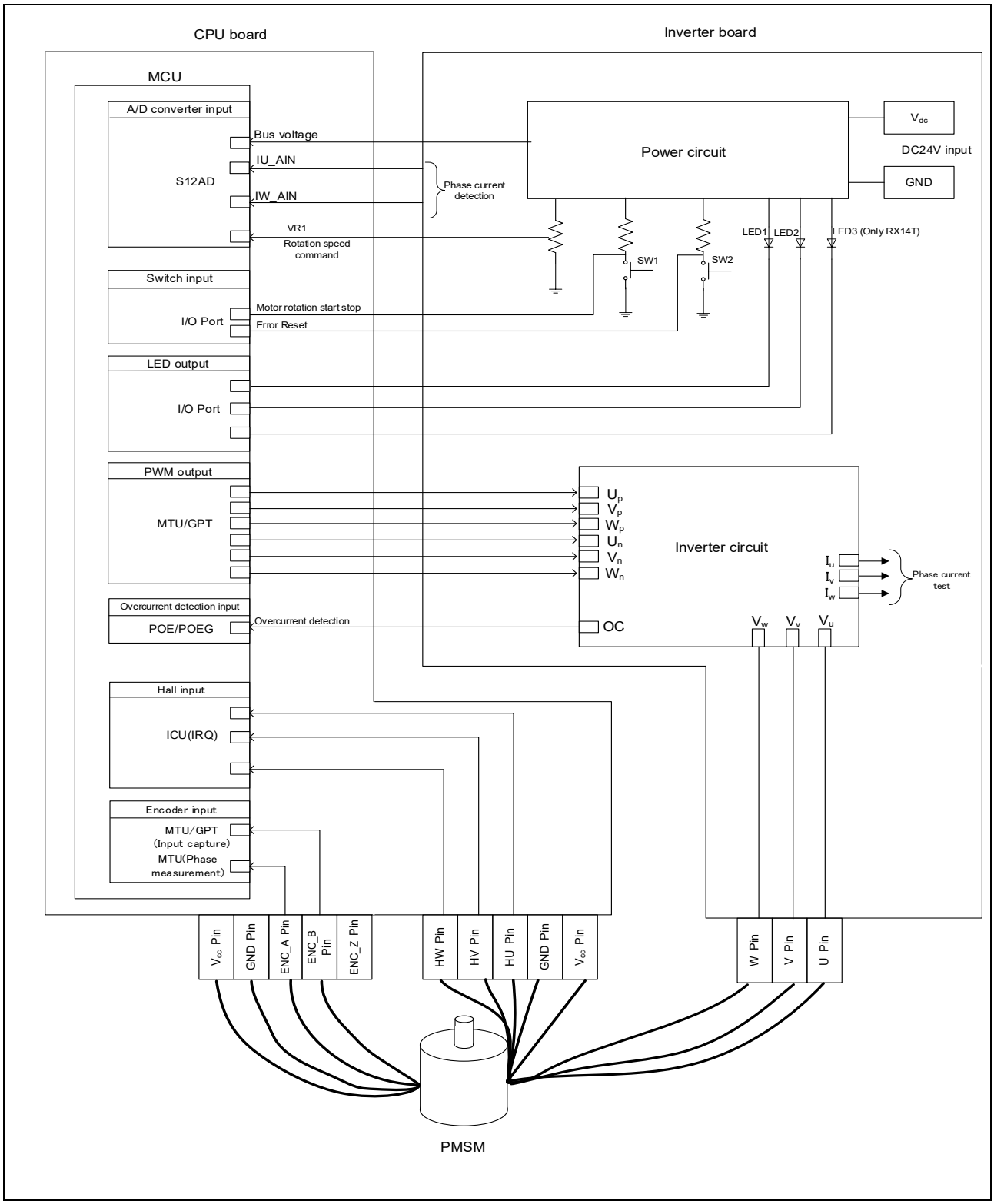


Figure 2-1 Hardware configuration diagram

2.2.2 Setting up RX26T/RX14T CPU board

This section describes how to install RX26T CPU board or RX14T CPU board, which can be plugged in to MCI-LV-1. You can plug RX26T/RX14T CPU board to the side of MCI-LV-1 board. A connector for writing the sample program, a connector for MC-COM, and a PG connector for an external encoder are also provided.

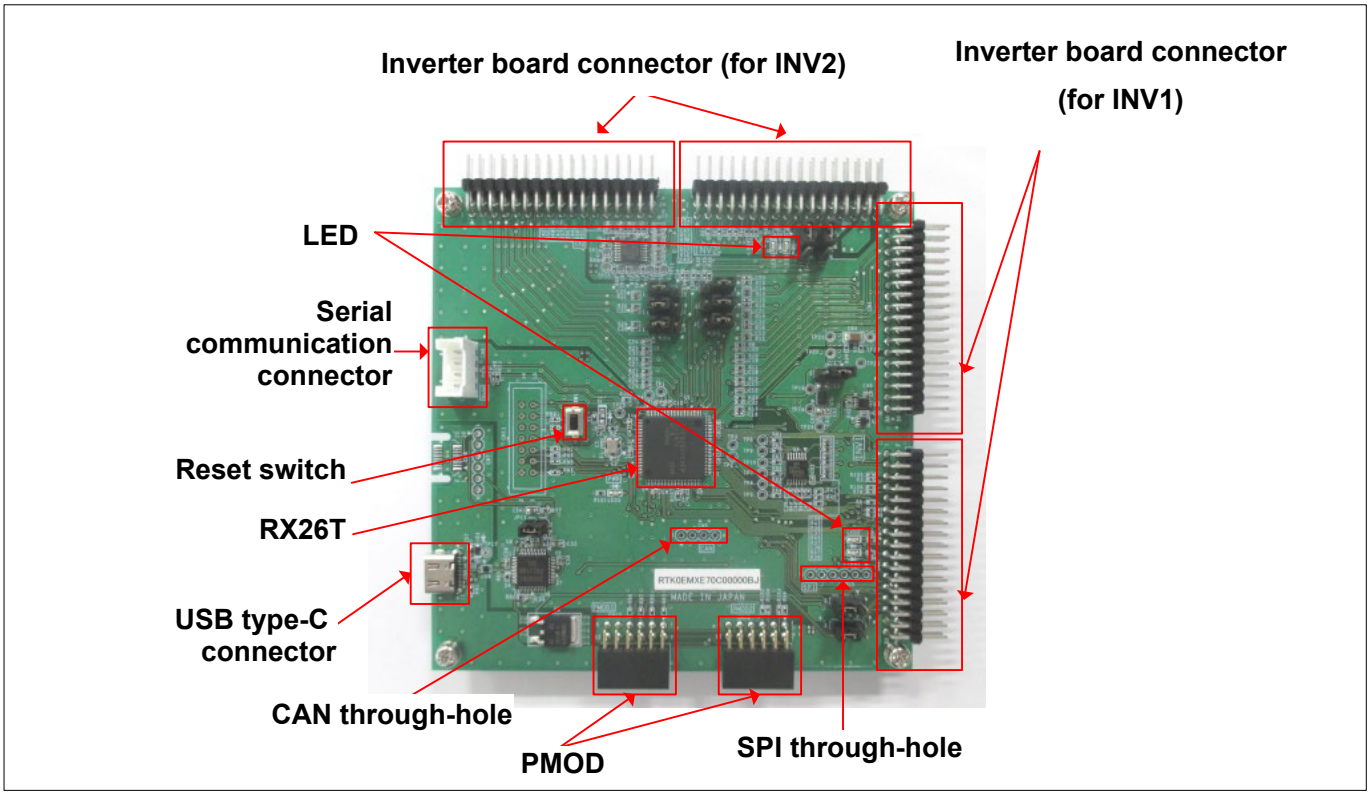


Figure 2-2 RX26T RAM64KB Version CPU board and its interface

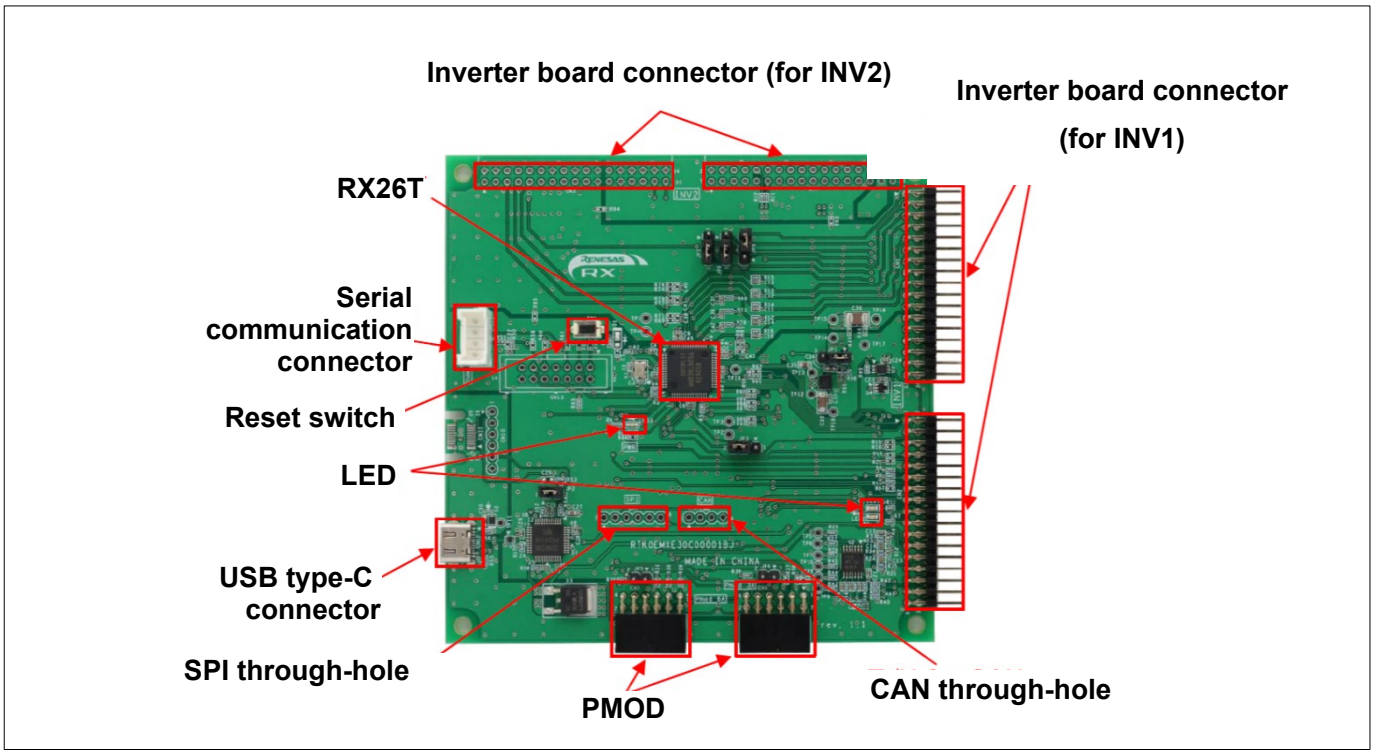


Figure 2-3 RX26T RAM48KB Version CPU board and its interface

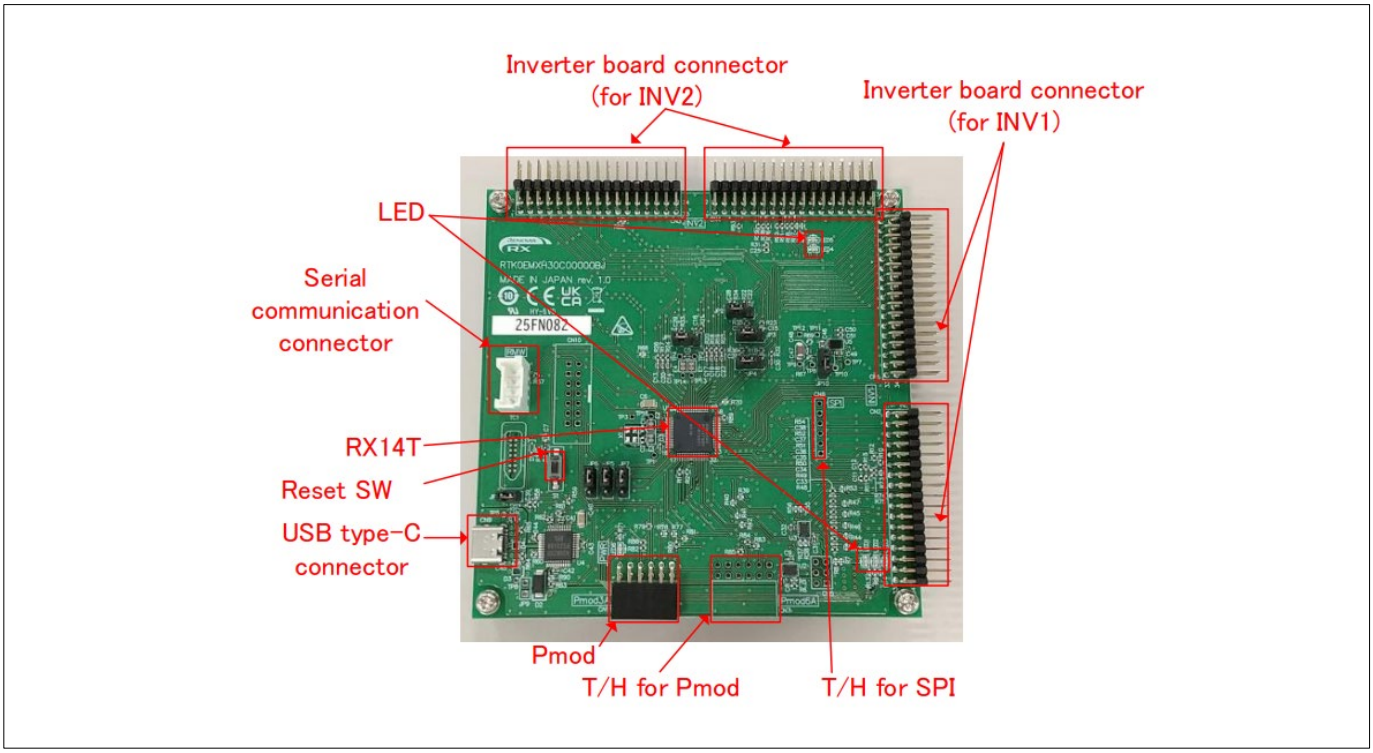


Figure 2-4 RX14T CPU board and its interface

2.2.3 On-board Debugger

This product is equipped with an on-board debugger circuit E2 On-Board (hereinafter referred to as E2OB), and the RX26T/RX14T program is rewritten using E2OB. When rewriting the program, open jumper JP11(RX26T RAM64KB Version), JP2(RX26T RAM48KB Version) or JP8(RX14T) and connect the CPU board to PC with a USB cable. E2OB acts as a debugger equivalent to E2 emulator Lite. When connecting from an integrated development environment (e.g., e^2 studio) or a flash programming tool (e.g., Renesas Flash Programmer), set the debugger (tool) type to "E2 emulator Lite" at the time of configuration.

After rewriting the program, short out JP11 ,JP2 or JP8 to enable the CPU board.

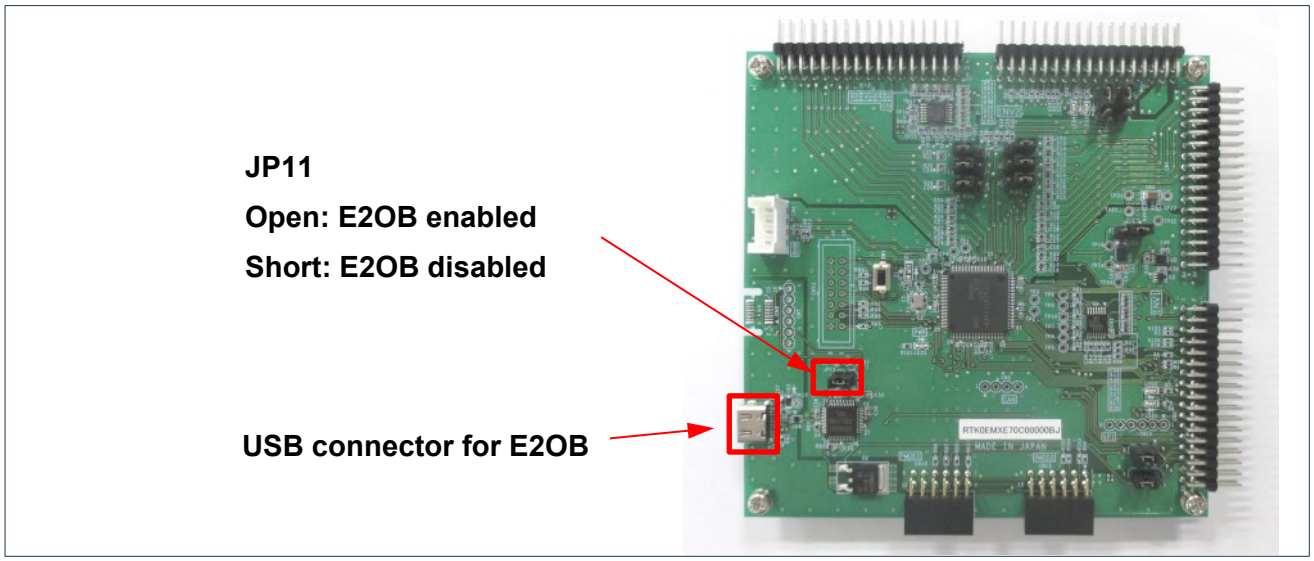


Figure 2-5 On-board debugger(RX26T RAM64KB Version)

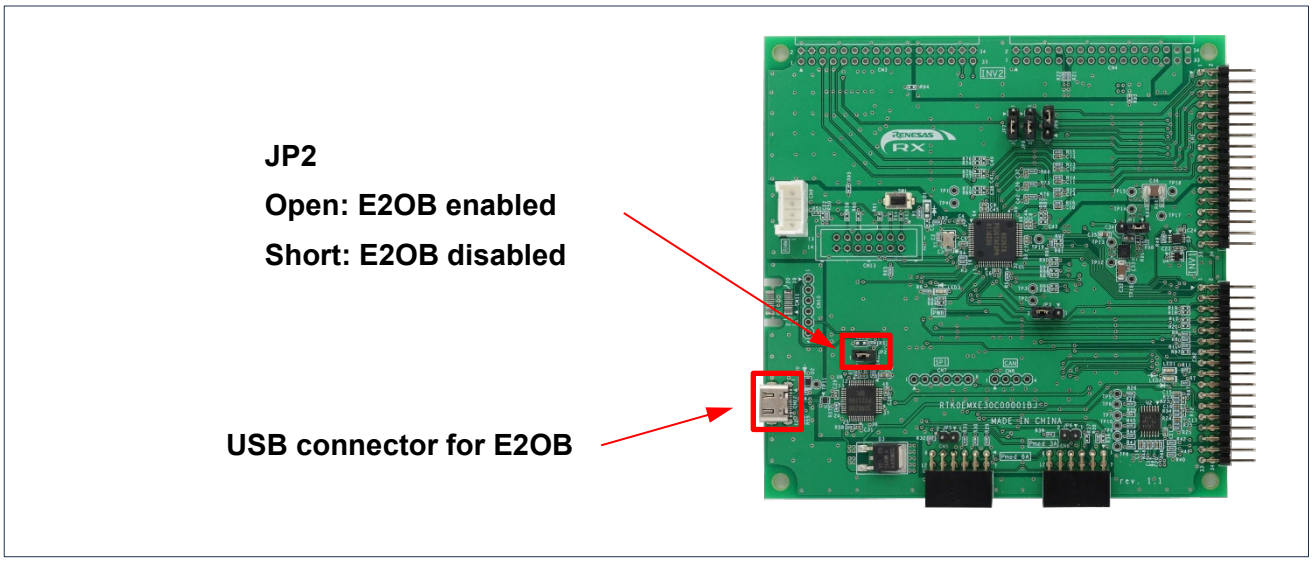


Figure 2-6 On-board debugger(RX26T RAM48KB Version)

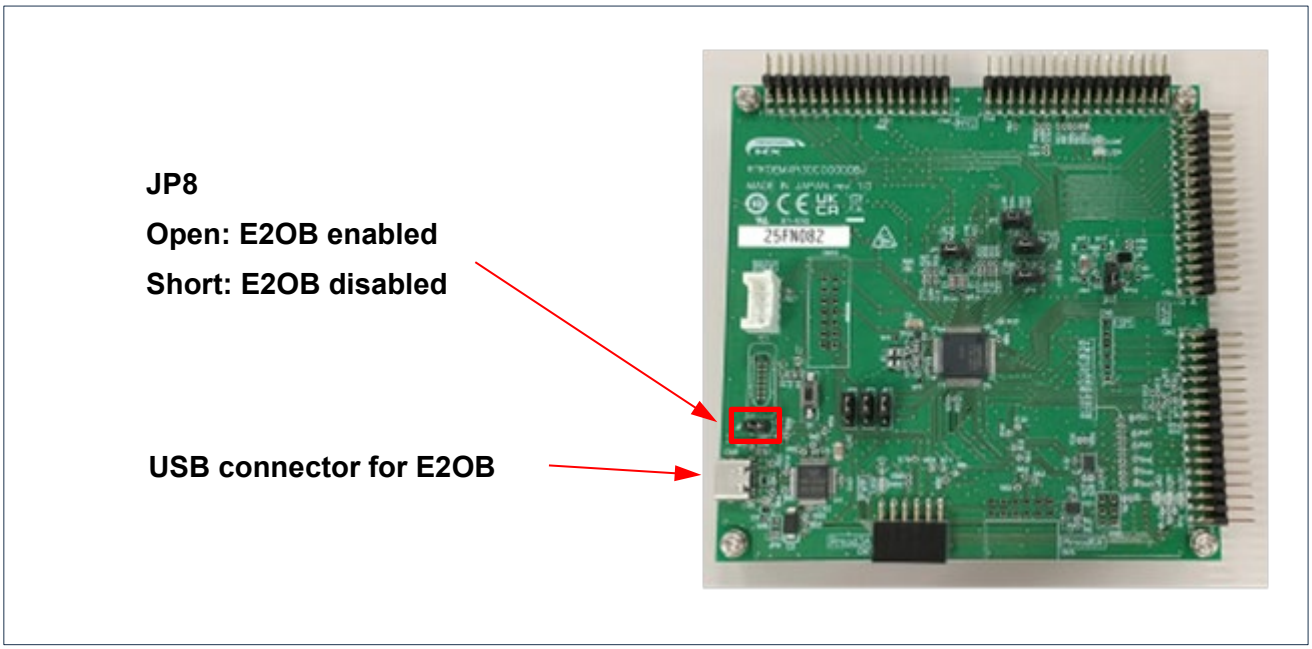


Figure 2-7 On-board debugger (RX14T)

2.2.4 Board user interface

Table 2-5 shows a list of user interfaces for this system.

Table 2-6 User Interface

Item	Interface parts	Function
Rotation position/speed	Variable resistor (VR1)	Rotation position/speed command value input (analog value)
START/STOP	Toggle switch (SW1)	Motor rotation start/stop changeover switch
ERROR RESET	Push switch (SW2)	Return command from error status
LED1	Orange LED	・When the motor is rotating: Lights up ・When stopped: Off
LED2	Orange LED	・When an error is detected: Lights up ・During normal operation: Off
LED3	Orange LED	Positioning completed: Lights up (Only RX14T)
RESET	Push button (SW1 on the CPU board)	System reset

2.2.5 Peripheral functions

Table 2-6 shows allocation of input/output functions and peripheral functions that are used in this system.

In the sample program, the peripheral functions are set using Smart configurator. See 5.14 for more information.

Table 2-7 Input/output functions and peripheral functions

Function	Peripheral functions		
	RX26T RAM64KB Version	RX26T RAM48KB Version	RX14T
Inverter bus voltage measurement	S12AD		
Rotation position/speed command value input (analog value)	S12AD		
START/STOP toggle switch	I/O Port (Input)		
LED1 on/off control	I/O Port(output)		
LED2 on/off control	I/O Port(output)		
	-		I/O Port(output)
U-phase current measurement	S12AD		
W-phase current measurement	S12AD		
PWM output (Up) / "High" active	MTU		
PWM output (Vp) / "High" active	MTU		
PWM output (Wp) / "High" active	MTU		
PWM output (Un) / "High" active	MTU		
PWM output (Vn) / “High” active	MTU		
PWM output (Wn) / “High” active	MTU		
Hall U phase input	ICU(IRQ)		
Hall V phase input	ICU(IRQ)		
Hall W phase input	ICU(IRQ)		
Encoder A phase input	MTU	GPT	MTU
Encoder B phase input	MTU	GPT	MTU
PWM emergency stop input when overcurrent is detected	POE		

3. Quick Start Guide

This chapter is a quick start guide for driving motors using the Renesas Flexible Motor Control Kit and sample programs. Please refer to the MCK-RX26T User's Manual (R12UZ0111) or MCK-RX14T User's Manual (R12UZ0192) for the board settings and connection of the Renesas Flexible Motor Control Kit. For details on how to use Renesas Motor Workbench, refer to the Renesas Motor Workbench User's Manual (R21UZ0004).

3.1 Downloading and writing sample programs

Download the sample program and write the program to the MCU on the CPU board using the IDE or Renesas Flash Programmer. Refer to the instruction manual of IDE and Renesas Flash Programmer on how to write the program.

When writing the program, it is necessary to set the JP11 ,JP2 or JP8 jumper on the CPU board to " E2OB enabled." For details on how to configure the JP11 JP2 or JP8 jumper, please refer to 2.2.3 On-board Debugger of this application note. Also, if you wish to run the software on the CPU board alone without using debug execution while connected to a PC after writing is complete, you must change the JP11,JP2 or JP8 jumper setting to " E2OB disabled "

3.2 Start Analyzer and RMT file

The motor control development support tool "Renesas Motor Workbench" is used as a user interface (rotation/stop command, rotation speed command, etc.). Please obtain the motor control development support tool "Renesas Motor Workbench" from our website.

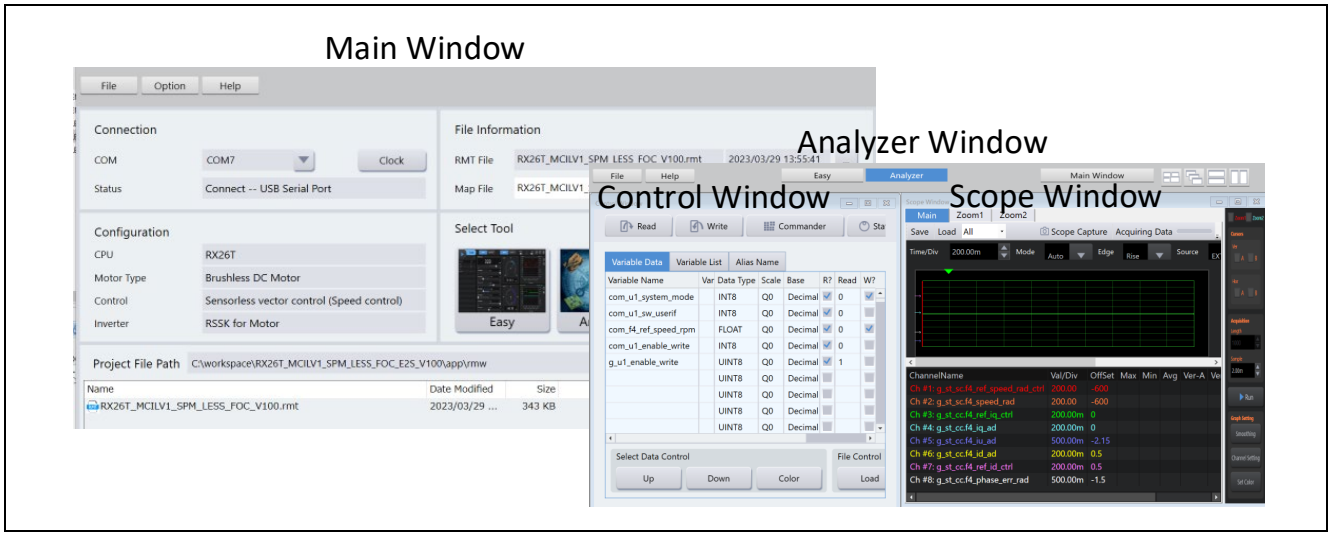


Figure 3-1 Appearance of Renesas Motor Workbench

How to use the motor control development support tool "Renesas Motor Workbench"

- Click the tool icon 

to start the tool.
- Select [File] → [Open RMT File (O)] from the MENU bar of the Main Panel. Load the RMT file in the "rmw" folder of the project folder.
- Select the COM of the kit connected by the COM of “Connection”.
- Click the “Analyzer” button on the “Select Tool” screen to display the Analyzer function screen.
- Drive the motor based on “RMW UI operation”. (For detail, see the section 3.4.)

What is an RMT file?

- An RMT file is a file that saves the environment information operated/set by RMW.
- By saving the environment information in the RMT file, you can recall the RMT file and restore the same environment.
- If the address information of the program is changed, load the Map file generated by building the program and save the RMT file again.

3.3 List of variables for Analyzer function

Table 3-1 shows a list of input variables when using the RMW UI user interface. When the same value as g_u1_enable_write is written to com_u1_enable_write, the input value to these variables is reflected in the corresponding variable in the motor module and used for motor control. However, variables marked with (*) do not depend on com_u1_enable_write.

Table 3-1 Analyzer Function List of main variables for input

Variable name for analyzer function input	Type	Contents
com_u1_sw_userif (*)	uint8_t	User interface switch 0: Use RMW user interface (default) 1: Use board user interface
com_u1_system_mode (*)	uint8_t	State management 0: Stop mode 1: Run mode 3: Reset
com_u1_ctrl_loop_mode	uint8_t	Control loop switching 0: Position control 1: Speed control (default)
com_f4_ref_position_deg (*)	int16_t	Position command value (machine angle) [degree]
com_f4_ref_speed_rpm (*)	int16_t	Speed command value (machine angle) [rpm]
com_u1_enable_write	uint8_t	Variable rewriting permission for user input Input data is reflected by variable matching with g_u1_enable_write

Next, Table 3-2 shows a list of the main structure variables that are often observed when performing drive evaluation of encoder position/velocity control. Please refer to it when displaying the waveform or reading the value of the variable with the Analyzer function. See 5.1.5 for more information on variables not in the list.

Table 3-2 List of main variables for encoder position/speed control

Encoder position/speed control main variable name	Type	Contents
g_st_encoder_vector.u2_error_status	uint16_t	Error status
g_st_cc.f4_id_ref	float	d-axis current command value [A]
g_st_cc.f4_id_ad	float	d-axis current detection value [A]
g_st_cc.f4_iq_ref	float	q-axis current command value [A]
g_st_cc.f4_iq_ad	float	q-axis current detection value [A]
g_st_cc.f4_iu_ad	float	U-phase current detection value [A]
g_st_cc.f4_iv_ad	float	V-phase current detection value [A]
g_st_cc.f4_iw_ad	float	W- phase current detection value [A]
g_st_cc.f4_vd_ref	float	d-axis voltage command value [V]
g_st_cc.f4_vq_ref	float	q-axis voltage command value [V]
g_st_cc.f4_refu	float	U-phase voltage command value [V]
g_st_cc.f4_refv	float	V-phase voltage command value [V]
g_st_cc.f4_refw	float	W-phase voltage command value [V]
g_st_sc.f4_ref_speed_rad_ctrl	float	Speed command value (machine angle) [rad/s]
g_st_sc.f4_speed_rad	float	Velocity detection value (machine angle) [rad/s]
g_st_pc.f4_ref_pos_rad_ctrl	float	Position command value (machine angle) [rad]
g_st_pc.f4_pos_rad	float	Position detection value (machine angle) [rad]

3.4 RMW UI operation

3.4.1 Analyzer operation example

The following is an example of operating the motor using the Analyzer function. The operation is performed in the “Control Window”. For details on the “Control Window”, refer to the “Renesas Motor Workbench User's Manual”.

Initially, the control loop is speed controlled. Perform the operation by referring to the following.

(a) Rotate the motor

(1) Make sure that "check" is entered in the [W?] field of "com_u1_system_mode " and "com_f4_ref_speed_rpm".
(2) Enter the command rotation speed in the [Write] field of "com_f4_ref_speed_rpm".
(3) Enter “1” in the [Write] field of “com_u1_system_mode_”.
(4) Press the “Write” button.

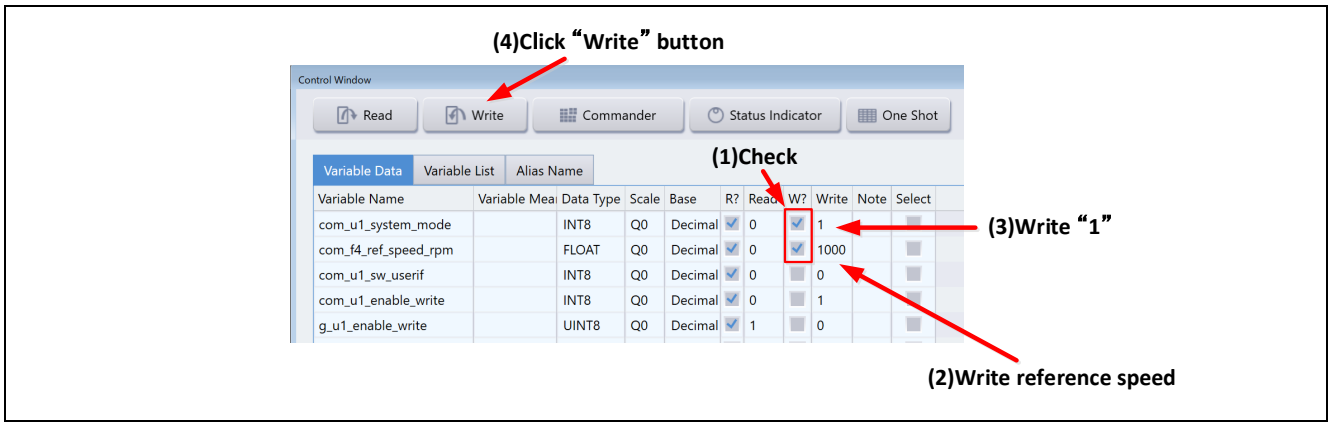


Figure 3-2 Motor rotation procedure

(b) Stop the motor

(1) Enter "0" in the [Write] field of "com_u1_system_mode ".
(2) Press the “Write” button.

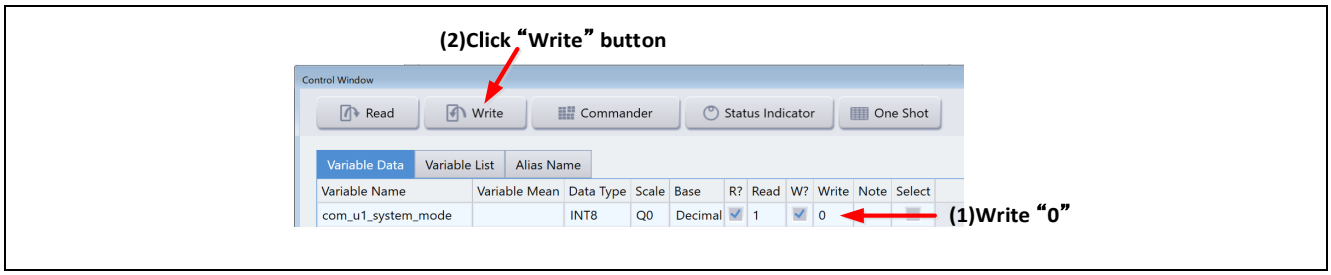


Figure 3-3 Motor stop procedure

(c) Processing when it stops (error)

(1) Enter "3" in the [Write] field of "com_u1_system_mode ".

(2) Press the “Write” button.

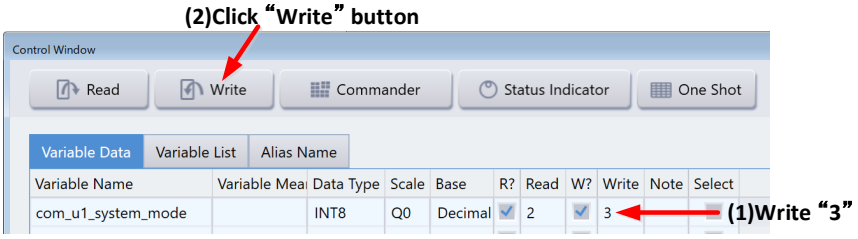


Figure 3-4 Error clearing procedure

3.4.2 User Button Function operation example

The following is an example of operating the motor using the User Button function. The user button shown as an example is included in the RMT file of the sample program.

- Drive/stop the motor by position control
  By setting as shown in Figure 3-5, the drive and stop are switched each time the button is pressed.

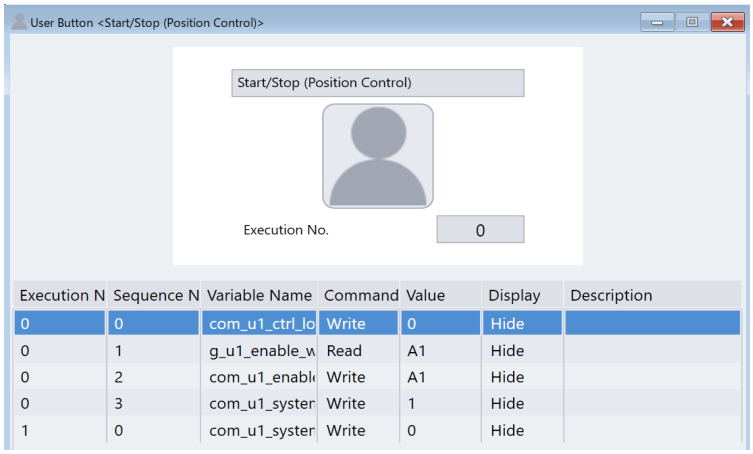


Figure 3-5 Motor drive/stop

- Changing the position command
  By setting as shown in Figure 3-6, you can change the position by inputting the position command and pressing the button.

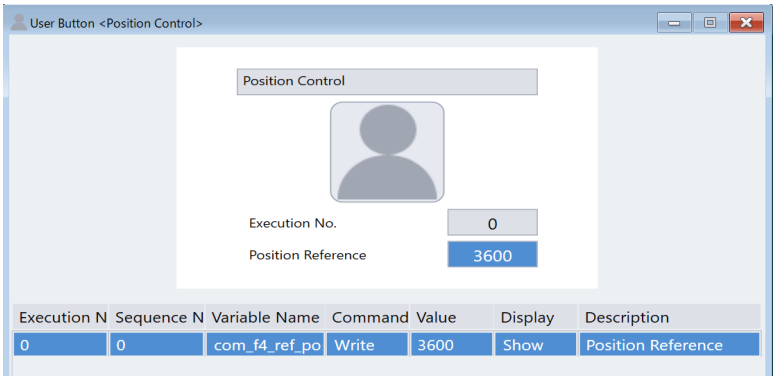


Figure 3-6 Change of position command

- Drive/stop the motor by speed control
  By setting as shown in Figure 3-7, the drive and stop are switched each time the button is pressed.

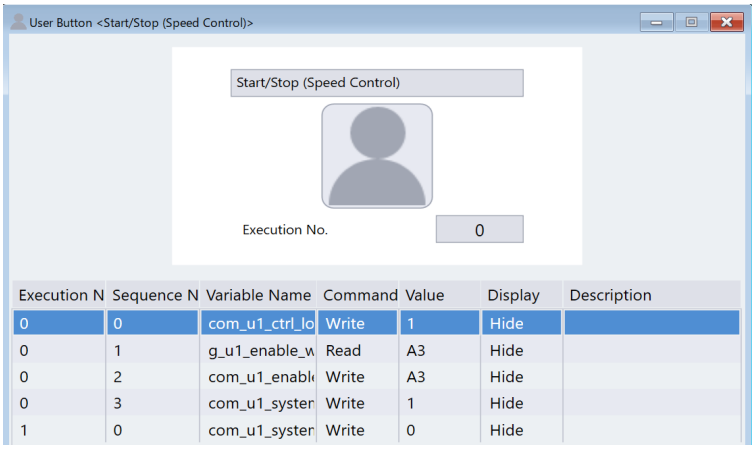


Figure 3-7 Motor drive/stop

- Changing the speed command
  By setting as shown in Figure 3-8, you can change the speed command by inputting the speed command and pressing the button.

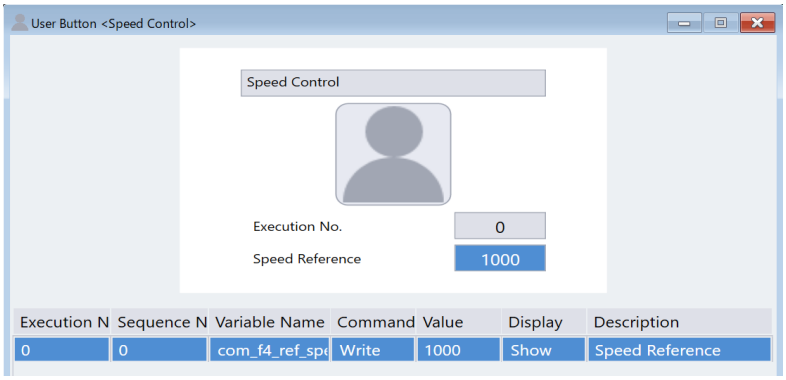


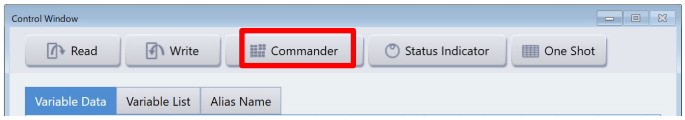
Figure 3-8 Changes of speed command

3.4.3 Commander function operation example

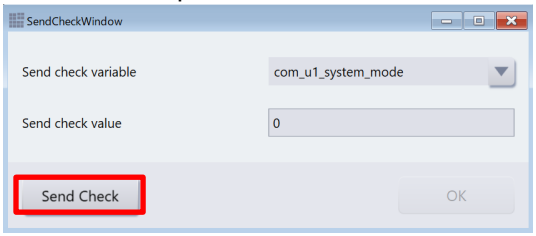
Position control using Commander function:

(Start Commander)

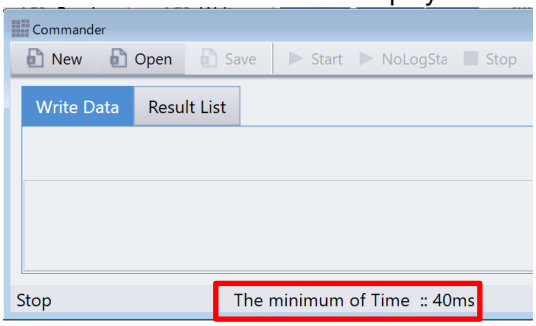
(1) Press the “Commander” button in the Control Window.



(2) The Commander window will open, then press the "Send Checker" button and check the data transmission speed.

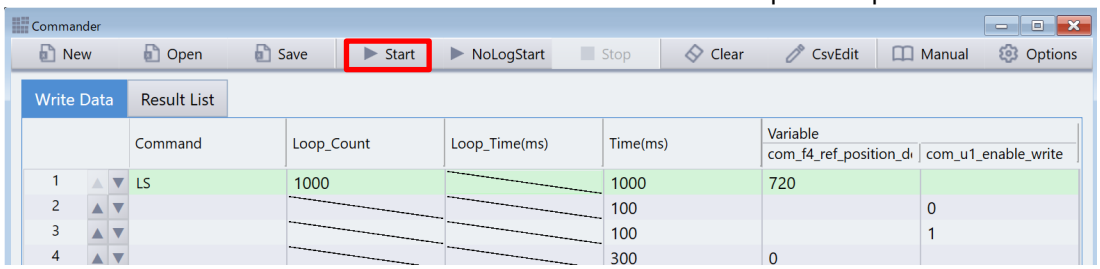


The minimum transfer time is displayed.



(3) Press the Open button to load “Position_test.csv”. Set to position control mode, write "1" to com_u1_system_mode and press the <Write> button to switch to run mode. The motor starts positioning control.

(4) Click the “Start” button in the Commander window to start the sequence operation.



3.5 Board UI operation

3.5.1 User interface switching

The downloaded sample program has RMW UI as the default setting. If you need to switch to the board UI, follow the steps below to switch the UI.

Make sure that "check" is checked in the [W?] Field of "com_u1_sw_userif", and enter "1" in the [Write] field. Press the "Write" button.

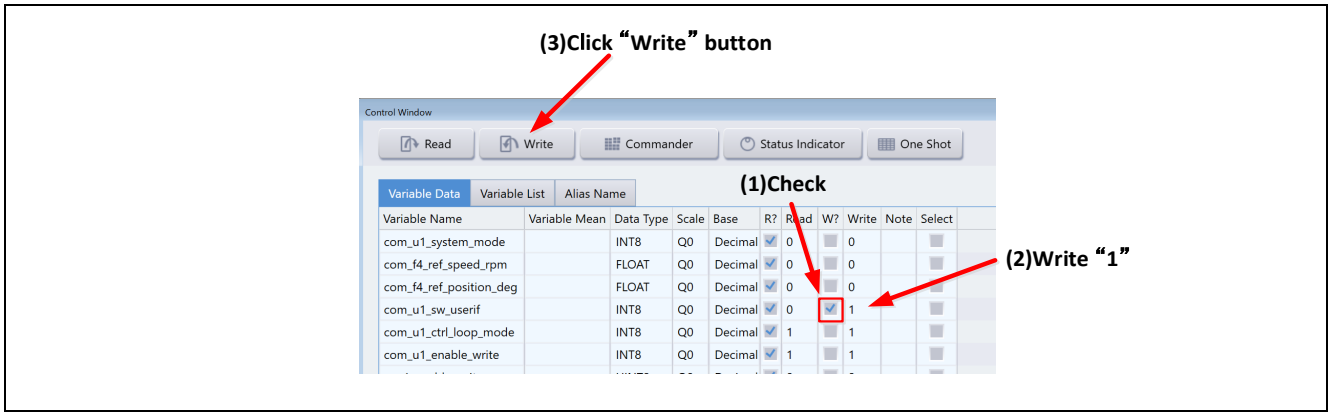


Figure 3-9 UI switching procedure

3.5.2 Motor start/stop

In the case of board UI, the start and stop of the motor is controlled by the input from SW1 of the inverter board (BOARD UI). A general purpose port is assigned to SW1. The terminal is read in the main loop, and when it is "ON" level, the start switch is considered to be pressed. When it is "OFF" level, the motor is considered to be stopped.

3.5.3 Motor rotation position/speed command value

The rotation position/speed command value of the motor is determined by A/D conversion of the output value (analog value) of VR1 of the inverter board. The A/D converted VR1 value is used as the rotation position/velocity command value as shown in the table below.

Table 3-3 Conversion ratio of rotation position/speed command value

Item	Conversion ratio (command value: A/D conversion value)	
Rotation position command value	CW	0 [degree] to 180[degree]: 07FFH to 0000H
	CCW	0 [degree] to -180[degree] :0800H to 0FFFH
Rotation speed command value	CW	0 [rpm] to 4000 [rpm]: 07FFH to 0000H
	CCW	0 [rpm] to -4000 [rpm]: 0800H to 0FFFH

4. Software

4.1 Software specification

The basic specifications of the software of this system are shown below.

Table 4-1 Encoder vector control software basic specifications

Item	Contents	
Control method	Vector control	
Motor control start/stop	Judgment based on SW1 level (”ON”: control start “OFF”: stop) or input from RMW	
Rotor pole position detection	Incremental encoder (A phase, B phase), Hall sensor (UVW phase)	
Input voltage	DC 24V	
Carrier frequency (PWM)	20 [kHz], Carrier cycle: 50 [µs]	
Dead time	RX26T: 2 [µs] RX14T: 1 [µs]	
Control cycle (current)	RX26T: 50 [µs] RX14T: 100 [µs] (number of thinning out 1 times)	
Control cycle (speed/position)	RX26T: 500 [µs] RX14T: 1 [ms]	
Position command value management	Board UI	Creation of position command value: Direct input by VR1 (Input range) -180° to +180°
	RMW UI	Creation of position command value: Position profile by velocity trapezoidal wave method (Input range) -32768 ° to 32767 ° (Speed limit) CW/CCW: -4000 to +4000 [rpm]
Speed command value management	CW: 0 [rpm] to 4000 [rpm] CCW: 0 [rpm] to -4000 [rpm]	
Position resolution	0.09 ° (encoder pulse: 1000 [p/r], 4000 [cpr] when multiplied by 4)	
Position dead zone *1	Encoder ± 1 count (± 0.09 °)	
Natural frequency of each control system	Current control system: 300 Hz Speed control system: 12 Hz Position control system: 4 Hz	
Compiler optimization settings	Optimization level	2 (-optimize = 2) (default setting)
	Optimization method	Code size-focused optimization (-size) (default setting)
Protection stop processing	Deactivate the motor control signal output (6 lines) under any of the following conditions. 1. Current in each phase exceeds 2.69 [A] (monitored every 50 [µs]) 2. Inverter bus voltage exceeds 60 [V] (monitored every 50 [µs]) 3. Inverter bus voltage less than 8 [V] (monitored every 50 [µs]) 4. Rotation speed exceeds 4500 [rpm] (monitored every 50 [µs]) 5. Hall sensor pattern error (at startup) When an external overcurrent detection signal (POE) and output short circuit are detected, set the PWM output terminal toInactive. .	

Note: 1. A dead zone is provided to prevent hunting during positioning.

4.2 Software configuration

The sample program consists of an application layer, a motor module, and a Smart configurator. The motor module controls by receiving instructions from the application layer operated by the user. The output to the HW layer is done via the Smart configurator.

4.2.1 Overall configuration

Figure 4-1 shows the overall configuration of the software.

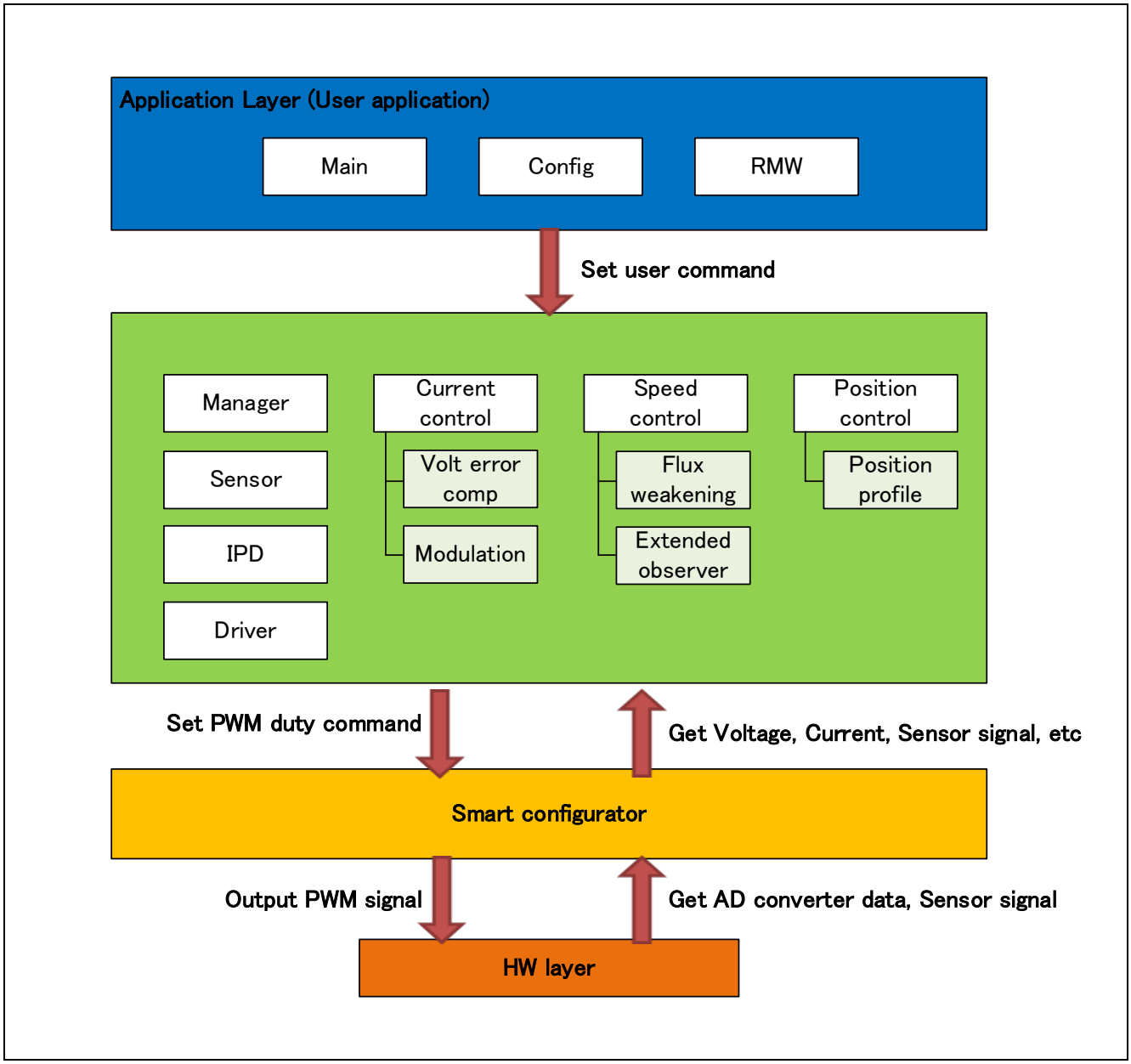


Figure 4-1 Overall configuration of motor control software

4.2.2 Motor module configuration

Figure 4-2 shows the configuration of the motor module. Table 4-2 gives an overview of each module. The manager module is an interface with other modules, and data is acquired and set in the appropriate module.

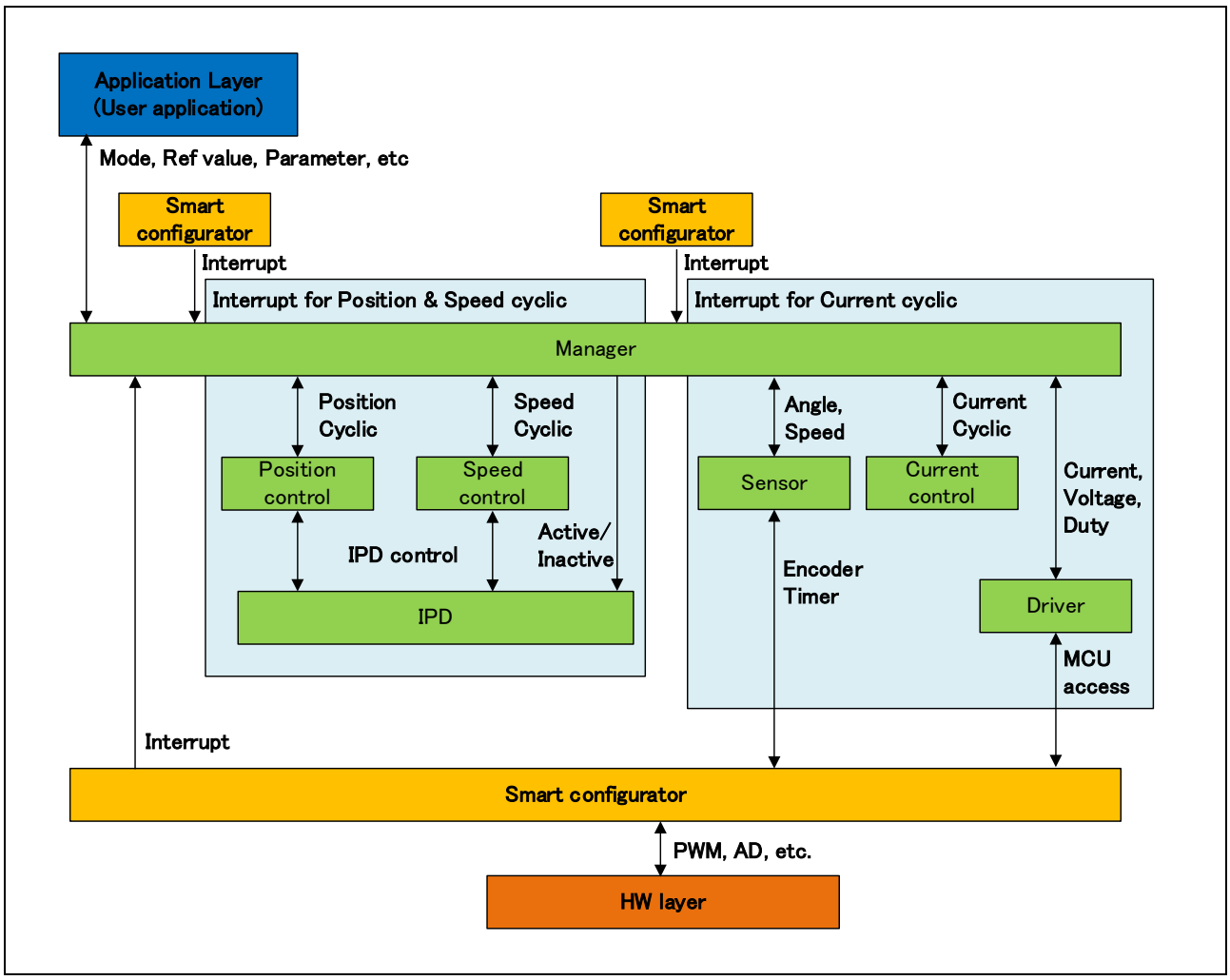


Figure 4-2 Motor module configuration

Table 4-2 Module overview

Module	Explanation	Detailed explanation
Application layer	Main processing, user area	5.1
Manager module	Management of the entire sample program and interface of each module	5.2
Current control module	Module for current control	5.3
Speed control module	Module for speed control	5.6
Position control module	Module related to position control	5.9
IPD module	Module for IPD control	5.11
Sensor module	Obtaining position/speed information from sensor signals Module	5.12
Driver module	Module for connection with Smart configurator	5.13
Smart configurator layer	Module for connecting to HW layer	5.14

4.3 File/folder structure

Table 4-3 shows the folder and file structure of the sample program.

Table 4-3 File/folder structure

Folder	Sub-folder	File	Remarks
app	main	r_app_main.c/h	User main function
	rmw	r_app_rmw.c/h	RMW Analyzer UI related function definition
		r_app_rmw_interrupt.c	RMW interrupt function definition
		RX26T: ICS2_RX26T.lib/h RX14T: ICS2_RX14T.lib/h	RMW communication library
	board_ui	r_app_board_ui.c/h	Board UI related function definition
		r_app_board_ui_ctrl.h	MCU-dependent board UI function definition
		RX26T:r_app_board_ui_ctrl_rx26T_mcilv1.c RX14T:r_app_board_ui_ctrl_rx14t_mcilv1.c	MCU-dependent board UI function definition
	cfg	r_app_control_cfg.h	App layer configuration definition
motor_module	encoder_vector_rx	r_motor_encoder_vector_action.c	Action function definition
		r_motor_encoder_vector_api.c/h	Manager module API function definition
		r_motor_encoder_vector_manager.c/h	Manager module local function definition
		r_motor_encoder_vector_protection.c/h	Function definition of protection function
		r_motor_encoder_vector_ statemachine.c/h	Function definition related to state transition
	current_rx	r_motor_current_api.c/h	Current control module API function definition
		r_motor_current.c/h	Current control module local function definition
		r_motor_current_modulation.c/h	Modulation module function definition
		r_motor_current_volt_err_comp.lib/h	Function definition of voltage error compensation module
		r_motor_current_pi_gain_calc.c	Current control module control gain calculation function definition
	speed_rx	r_motor_speed_api.c/h	API function definition of speed control module
		r_motor_speed.c/h	Local function definition of speed control module
		r_motor_speed_fluxwkn.lib/h	Function definition of weak flux module
		r_motor_speed_extobserver.lib/h	Function definition of the disturbance observer module
		r_motor_speed_pi_gain_calc.c	Calculation of control gain of speed control module
	position_rx	r_motor_position_api.c/h	Position control module API function definition
		r_motor_position.c/h	Position control module local function definition
		r_motor_position_profiling.c/h	Function definition for position control command value creation
		r_motor_position_gain_calc.c	Position control module control gain calculation function definition
	ipd_rx	r_motor_ipd_api.lib/h	IPD module API function definition

Folder	Sub-folder	File	Remarks
motor_module	driver_rx	r_motor_driver.c/h	Driver module function definition
	sensor_rx	r_motor_sensor_api.c/h	Sensor module API function definition
		r_motor_sensor_encoder.c/h	Sensor module encoder processing function definition
		r_motor_sensor_hall.c/h	Hall sensor processing function definition for sensor module
	general	r_motor_filter.c/h	General-purpose filter function definition
		r_motor_pi_control.c/h	PI control function definition
		r_motor_common.h	Common definition
	cfg	r_motor_inverter_cfg.h	Inverter configuration definition
		r_motor_module_cfg.h	Control module configuration definition
		r_motor_targetmotor_cfg.h	Motor configuration definition
		r_mtr_control_parameter.h	Tuning Result by the Tuner functions of RMW*1 (Control parameters definition)
		r_mtr_motor_parameter.h	Tuning Result by the Tuner functions of RMW*1 (Motor parameters definition)
QE_Motor			Generated files by QE for Motor
src	smc_gen	See the next table	Drivers and APIs generated by Smart configurator

Note: 1. In case the tuning function of QE for Motor is executed, the files will be updated.

Peripheral function drivers can be easily generated by using the smart configurator. You can also generate drivers for the multifunction timer pulse unit and 12-bit A/D converter by using the components dedicated to the motor.

Smart configurator saves and refers to the setting information such as the microcontroller, peripheral functions, and terminal functions used in the project in the project file (* .scfg). To check the peripheral function settings of this software, refer to the following file.

"RX26T_xxx_MCILV1_SPM_ENCD_FOC_yyy_Vzzz.scfg"
(xxx: MCBA means for RX26T RAM64KB Version, MCBC means for RX26T RAM48KB Version.
yyy: CSP means CS + version, E2S means e^2 studio version. zzz: revision number)

"RX14T_MCILV1_SPM_ENCD_FOC_yyy_Vzzz.scfg"
(In the above file name, the "yyy" portion indicates the edition: CSP indicates the CS+ edition and E2S indicates the e^2 studio edition. The "zzz" portion indicates the revision number.)

The folders and file structure generated by smart configurator are shown below.

Table 4-4 Smart Configurator folder and file structure (RX26T RAM64KB Version)

Folder	Sub-folder	Sub-folder 2	File	Remarks
src	smc_gen	Config_ICU	Config_ICU.c/h	Hall interrupt controller related function definition
			Config_ICU_user.c	Hall interrupt controller related user function definition
		Config_S12AD2	Config_S12AD2.c/h	12bit ADC related function definition
			Config_S12AD2_user.c	12bit ADC related user function definition
		Config_PORT	Config_PORT.c/h	Port-related function definition
			Config_PORT_user.c	Port-related user function definition
		Config_CMT0	Config_CMT0.c/h	CMT related function definition for control cycle
			Config_CMT0_user.c	CMT related user function definition for control cycle
		Config_GPT3	Config_GPT3.c/h	GPT3 related function definition for speed measurement
			Config_GPT3_user.c	GPT3 related user function definition for speed measurement
		Config_IWDT	Config_IWDT.c/h	IWDT related function definition
			Config_IWDT_user.c	IWDT related user function definition
		Config_MTU0	Config_MTU0.c/h	Definition of MTU related functions for speed measurement
			Config_MTU0_user.c	MTU related user function definition for speed measurement
		Config_MTU1	Config_MTU1.c/h	MTU related function definition for phase factor
			Config_MTU1_user.c	MTU related user function definition for phase factor
		Config_POE	Config_POE.c/h	POE related function definition
			Config_POE_user.c	Related user function definition
		Config_MOTOR	Config_MOTOR.c/h	Multi-function timer and AD conversion related function for motor control
			Config_MOTOR_user.c	Multi-function timer and AD conversion related user function for motor control

Table 4-5 Smart Configurator folder and file structure (RX26T RAM64KB Version)

Folder	Sub-folder	Sub-folder 2	File	Remarks
src	smc_gen	Config_ICU	Config_ICU.c/h	Hall interrupt controller related function definition
			Config_ICU_user.c	Hall interrupt controller related user function definition
		Config_S12AD2	Config_S12AD2.c/h	12bit ADC related function definition
			Config_S12AD2_user.c	12bit ADC related user function definition
		Config_PORT	Config_PORT.c/h	Port-related function definition
			Config_PORT_user.c	Port-related user function definition
		Config_CMT0	Config_CMT0.c/h	CMT related function definition for control cycle
			Config_CMT0_user.c	CMT related user function definition for control cycle
		Config_CMTW0	Config_CMTW0.c/h	CMTW0 related function definition for speed measurement
			Config_CMTW0_user.c	CMTW0 related user function definition for speed measurement
		Config_IWDT	Config_IWDT.c/h	IWDT related function definition
			Config_IWDT_user.c	IWDT related user function definition
		Config_MTU0	Config_MTU0.c/h	Definition of MTU related functions for speed measurement
			Config_MTU0_user.c	MTU related user function definition for speed measurement
		Config_ELC	Config_ELC.c/h	Definition of ELC functions for speed measurement
			Config_ELC_user.c	ELC user function definition for speed measurement
		Config_GPT5	Config_GPT5.c/h	GPT related function definition for phase factor
			Config_GPT5_user.c	GPT related user function definition for phase factor
		Config_POE	Config_POE.c/h	POE related function definition
			Config_POE_user.c	Related user function definition
		Config_MOTOR	Config_MOTOR.c/h	Multi-function timer and AD conversion related function for motor control
			Config_MOTOR_user.c	Multi-function timer and AD conversion related user function for motor control

Table 4-6 Smart Configurator folder and file structure (RX14T)

Folder	Sub-folder	Sub-folder 2	File	Remarks
src	smc_gen	Config_ICU	Config_ICU.c/h	Hall interrupt controller related function definition
			Config_ICU_user.c	Hall interrupt controller related user function definition
		Config_S12AD1	Config_S12AD1.c/h	12bit ADC related function definition
			Config_S12AD1_user.c	12bit ADC related user function definition
		Config_PORT	Config_PORT.c/h	Port-related function definition
			Config_PORT_user.c	Port-related user function definition
		Config_CMT0	Config_CMT0.c/h	CMT related function definition for control cycle
			Config_CMT0_user.c	CMT related user function definition for control cycle
		Config_IWDT	Config_IWDT.c/h	IWDT related function definition
			Config_IWDT_user.c	IWDT related user function definition
		Config_MTU0	Config_MTU0.c/h	Definition of MTU related functions for speed measurement
			Config_MTU0_user.c	MTU related user function definition for speed measurement
		Config_MTU1	Config_MTU1.c/h	MTU related function definition for phase factor
			Config_MTU1_user.c	MTU related user function definition for phase factor
		Config_GPT0	Config_GPT0.c/h	Definition of GPT related functions for speed measurement
			Config_GPT0_user.c	GPT related user function definition for speed measurement
		Config_POE	Config_POE.c/h	POE related function definition
			Config_POE_user.c	Related user function definition
		Config_MOTOR	Config_MOTOR.c/h	Multi-function timer and AD conversion related function for motor control
			Config_MOTOR_user.c	Multi-function timer and AD conversion related to user function for motor control

In addition to each of the above tables, 4 folders are automatically generated when using smart configurator.

r_bsp: Contains various BSP (Board Support Package) files. See the “readme.txt” file in the “r_bsp” folder for details.

general: Contains various files commonly used by smart configurator generation drivers.

r_config: Contains the configuration header files for the driver initialization function with the names MCU package, clock, interrupt, and R_xxx_Open.

r_pincfg: Contains various files related to pin settings.

5. Functionality

5.1 Application layer

The application layer is used for selecting the user interface (UI), setting command values for controlling motor modules that use RMW, and updating parameters for control modules. Two user interfaces (configured and processed in the sample program) are used: the Board UI, which uses the switches and variable resistor on the inverter board to drive the motor, and the RMW UI, which uses RMW to drive the motor. These UIs are also used to control whether to drive or stop the motor and to set control command values.

5.1.1 Functions

Table 5-1 lists the functions that are configured in the application layer.

Table 5-1 Functions available in the application layer

Function	Description
Main processing	Enables or disables each user command in the system.
UI processing	Selects and switches between the Board UI and RMW UI.
Board UI processing	Obtains and sets command values for position control and speed control.
RMW UI processing	Acquires and sets parameters (including command values for speed and position information).

5.1.2 Module configuration diagram

Figure 5-1 shows the module configuration.

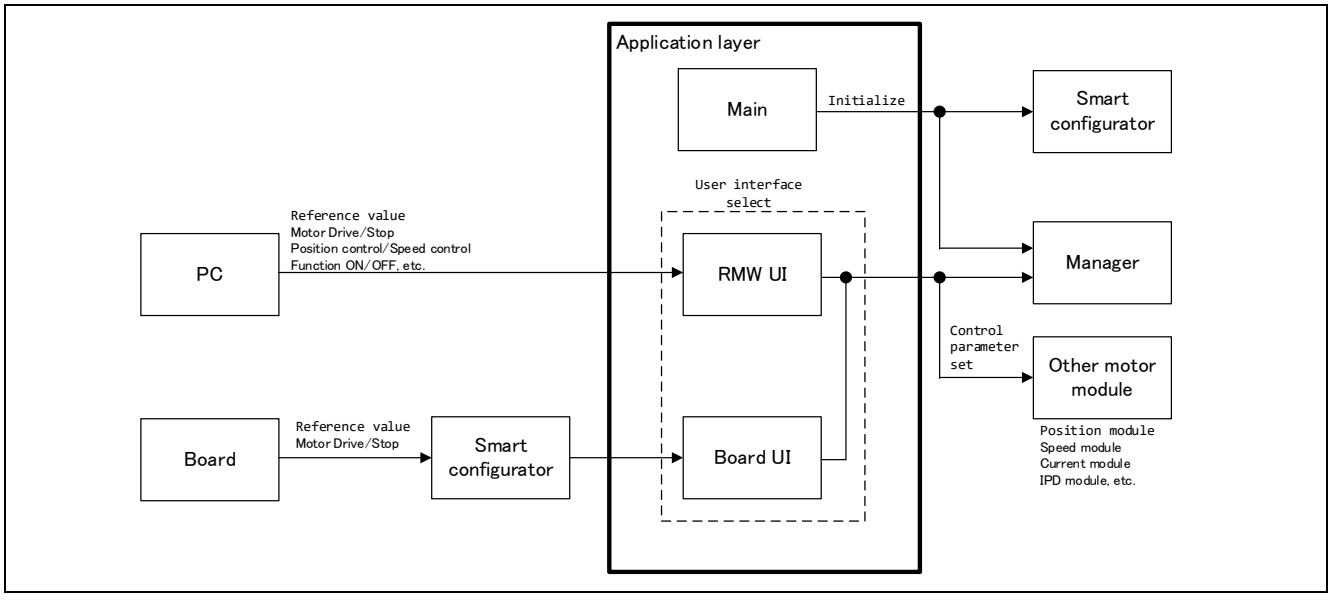


Figure 5-1 Configuration of the application layer

5.1.3 Flowchart

5.1.3.1 Main processing

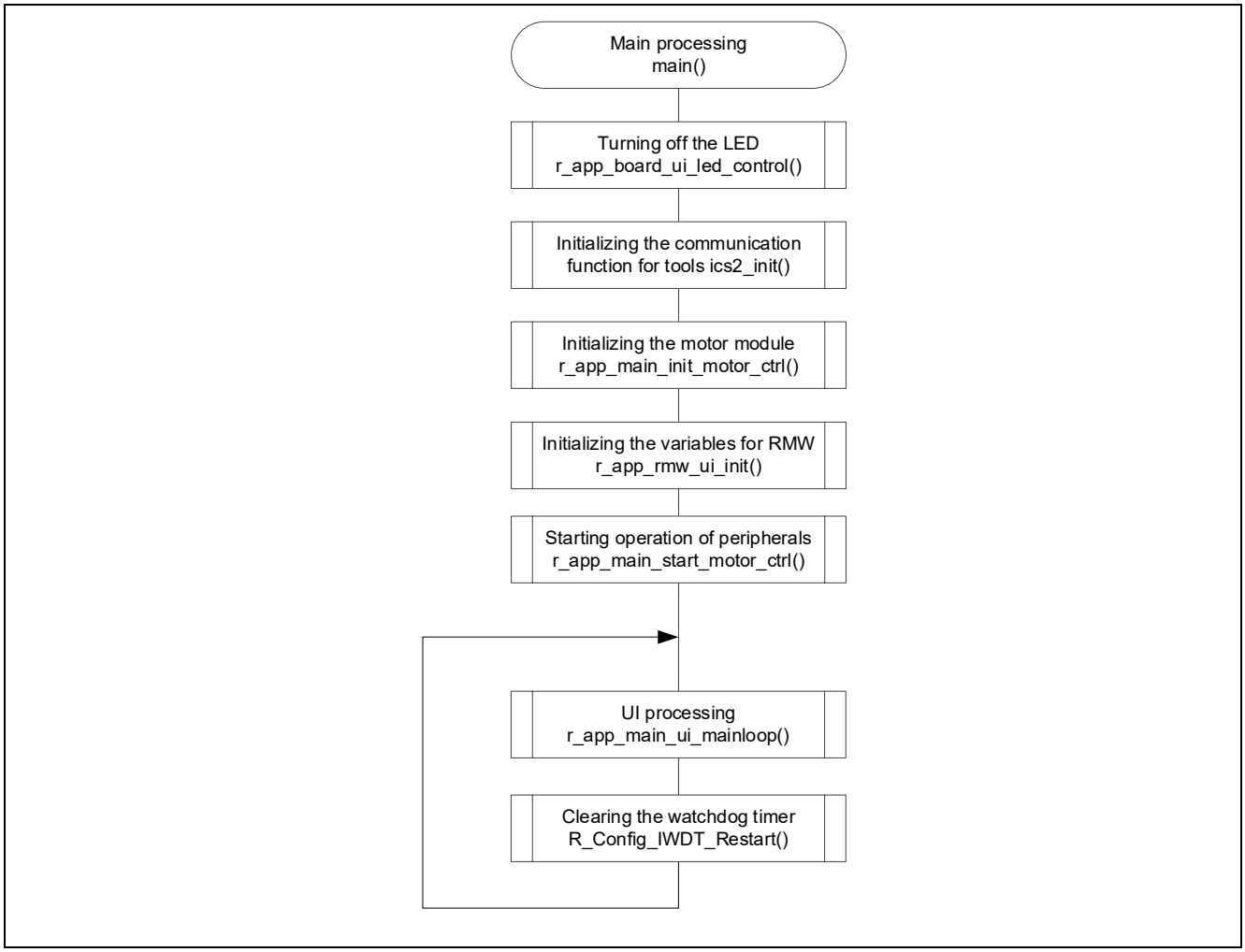


Figure 5-2 Flowchart for the main processing

5.1.3.2 UI processing

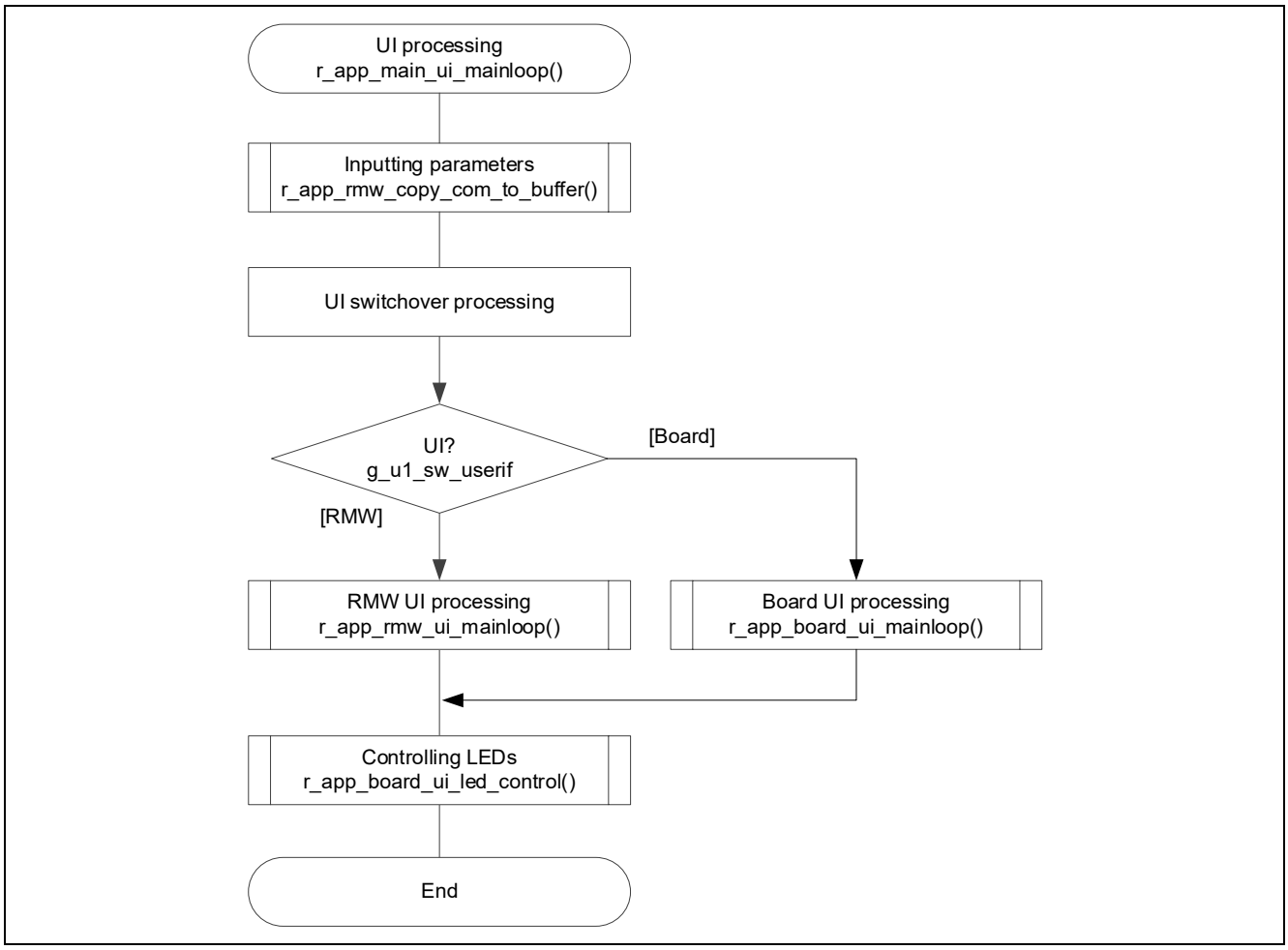


Figure 5-3 Flowchart for the UI processing

5.1.3.3 Board UI processing

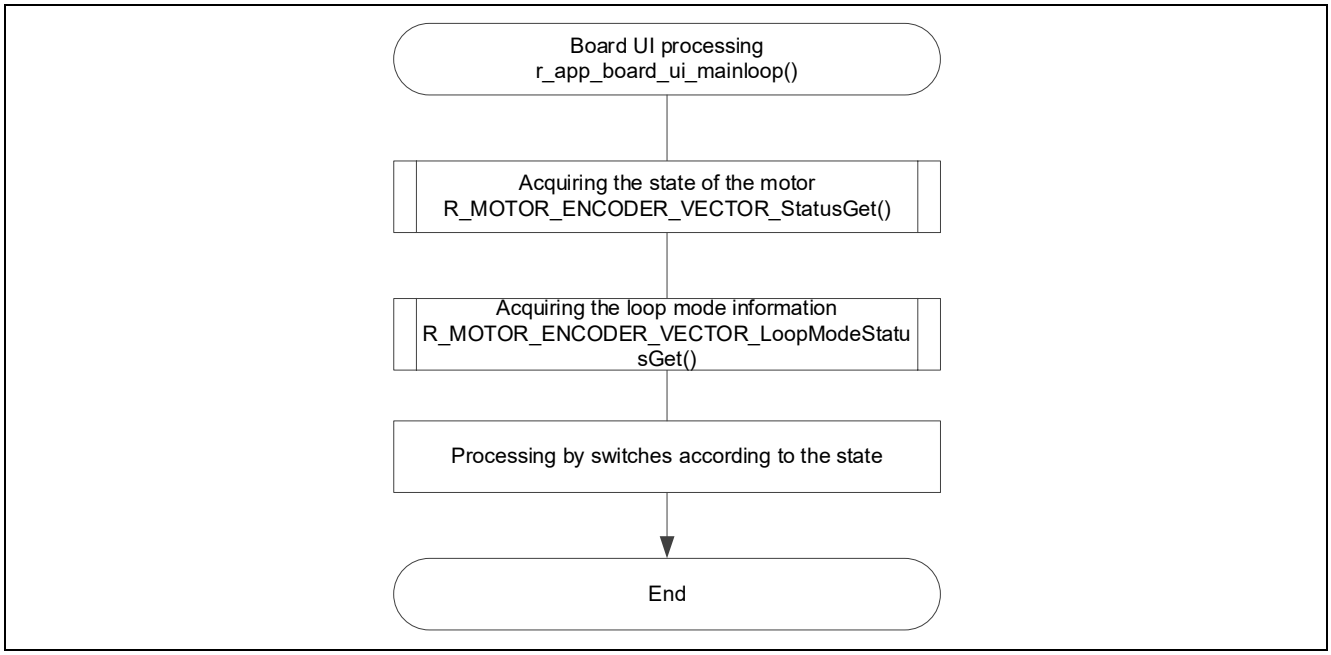


Figure 5-4 Flowchart for the Board UI processing

5.1.3.4 RMW UI processing

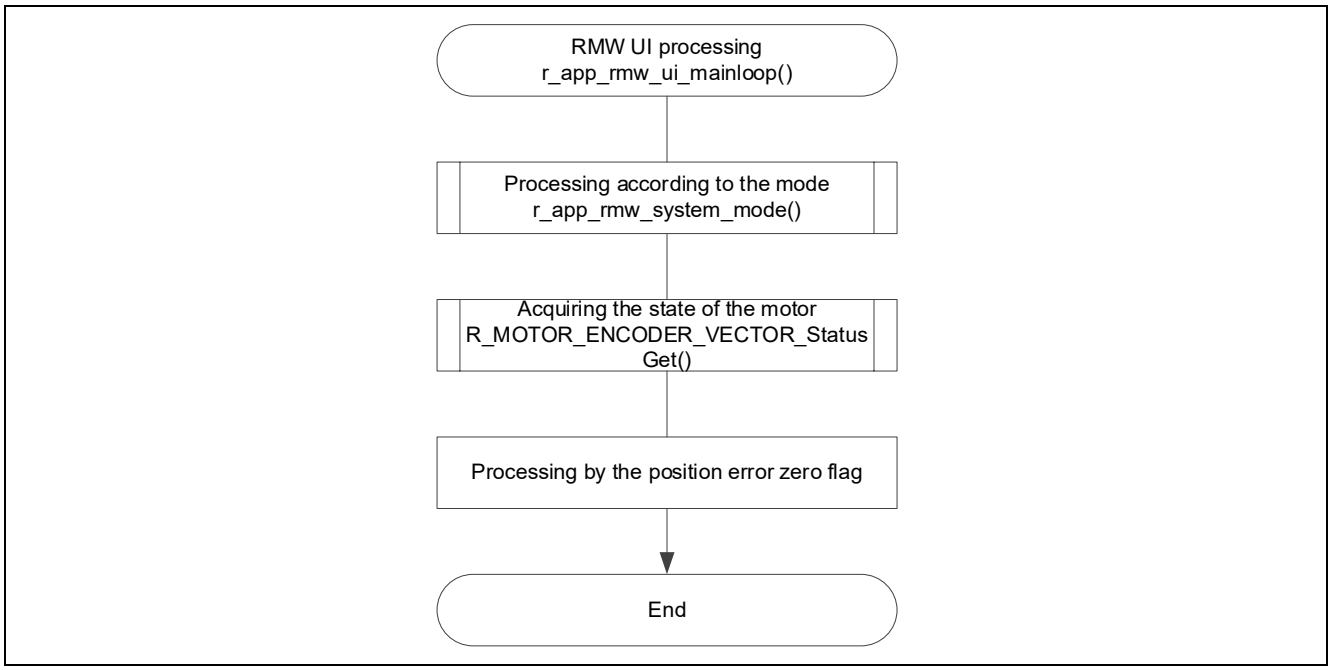


Figure 5-5 Flowchart for the RMW UI processing

5.1.4 Configurations

Table 5-2 shows the configurations used in the application layer.

Table 5-2 List of configurations

File name	Macro name	Description
r_app_control_cfg.h	APP_CFG_USE_UI	Initial UI setting RMW: MAIN_UI_RMW Board: MAIN_UI_BOARD
	APP_CFG_FREQ_BAND_LIMIT	This item sets the limit value for maintaining separation between the natural frequencies for current control, speed control, and position control.
	APP_CFG_MAX_CURRENT_OMEGA	This item sets the upper limit on the natural frequencies for current control [Hz].
	APP_CFG_MIN_OMEGA	This item sets the lower limit on natural frequencies [Hz].
	APP_CFG_SCI_CH_SELECT	This item is used to select the SCI channel for RMW.

Table 5-3 Configuration information initial value list

Macro name	Settings		
	RX26T RAM64KB Version	**RX26T RAM48KB Version**	**RX14T**
APP_CFG_USE_UI	MAIN_UI_RMW		
APP_CFG_FREQ_BAND_LIMIT	3.0f		
APP_CFG_MAX_CURRENT_OMEGA	1000.0f		
APP_CFG_MIN_OMEGA	1.0f		
APP_CFG_SCI_CH_SELECT	0x60 (SCI6)	0x10(SCI1)	0x50 (SCI5)

5.1.5 Structure and variable information

Table 5-4 lists the variables that can be used by users in the application layer. Table 5-5 lists the members of the structure provided for updating the motor module parameters by using RMW.

Table 5-4 List of variables

Variable	Description
g_st_rmw_input_buffer	Structure for updating the RMW variables
g_u1_update_param_flag	Buffer transfer completion flag
com_u1_system_mode	Variable to switch the system mode for user entry 0: Stopping the motor 1: Driving the motor 3: Canceling the error
g_u1_system_mode	System mode 0: Motor stop 1: Motor drive 2: Error
com_u1_enable_write	Whether to enable rewrite of variables for user entry
g_u1_enable_write	Whether to enable rewrite of variables
com_u1_sw_userif	Variable to switch the UI for user entry 0: RMW UI 1: Board UI
g_u1_sw_userif	Variable to switch the UI
com_u2_offset_calc_time	Current offset value calculation time
com_u2_mtr_pp	Number of pole pairs of the motor to be driven
com_f4_mtr_r	Resistance of the motor to be driven [Ω]
com_f4_mtr_ld	d-axis inductance of the motor to be driven [H]
com_f4_mtr_lq	q-axis inductance of the motor to be driven [H]
com_f4_mtr_m	Magnetic flux of the motor to be driven [Wb]
com_f4_mtr_j	Rotor inertia of the motor to be driven [kgm^2]
com_f4_nominal_current_rms	Rated current of the motor to be driven [Arms]
com_f4_max_speed_rpm	Maximum speed (mechanical angle) of the motor to be driven [rpm]
com_u1_ctrl_loop_mode	Switching of the control loop 0: Position control 1: Speed control
com_u1_encd_angle_adj_mode	Magnetic pole position detection mode 0: Position detection by forced excitation 1: Position detection using a Hall sensor
com_u2_encd_cpr	Number of encoder pulses [p/r]
com_f4_hs_change_speed_rpm	Switching speed (switching function for speed calculation at high speed) [rpm]
com_f4_hs_change_margin_rpm	Switching speed margin (switching function for speed calculation at high speed) [rpm]
com_f4_ol_ref_id	d-axis current command value [A]

Variable	Description
com_f4_id_up_time	Additional time for the d-axis current command value
com_f4_current_omega_hz	Natural frequency for current control [Hz]
com_f4_current_zeta	Attenuation coefficient for current control
com_f4_speed_omega_hz	Natural frequency for speed control [Hz]
com_f4_speed_zeta	Attenuation coefficient for speed control
com_f4_speed_lpf_hz	Speed LPF cut-off frequency [Hz]
com_f4_ref_speed_rpm	Speed command value (mechanical angle) [rpm]
com_f4_speed_rate_limit_rpm	Maximum increment/decrement width for the speed command [rpm/s] (used when speed control is enabled)
com_f4_overspeed_limit_rpm	Speed limit value (mechanical angle) [rpm]
com_u1_pos_cmd_mode	Switching of the entry method for the position command value 0: Position always-0 command 1: Step response 2: Trapezoidal wave response
com_u2_pos_interval_time	Stationary wait time for position response [s]
com_u2_pos_dead_band	Dead band (number of encoder pulses) [pulses]
com_u2_pos_band_limit	Position error zero range [pulses]
com_f4_pos_omega_hz	Natural frequency for position control [Hz]
com_f4_pos_ff_ratio	Position feed-forward gain
com_f4_ref_position_deg	Position command value (mechanical angle) [degrees]
com_u1_flag_extobserver_use	Disturbance torque/speed estimation observer setting 0: Disabled 1: Enabled
com_f4_extobs_omega	Natural frequency for speed control module observer [Hz]
com_f4_accel_time	Acceleration time [s] (for creating the position command value)
com_f4_posprof_max_speed_rpm	Maximum speed value for the position profile (mechanical angle) [rpm]
com_u1_flag_ipd_use	IPD module setting 0: Disabled 1: Enabled
com_f4_ipd_speed_k_ratio	Speed gain ratio during IPD control
com_f4_ipd_pos_kp_ratio	Position P/control amount ratio during IPD control
com_f4_ipd_omega_hz	Natural frequency for IPD control
com_f4_ipd_pos_ff_ratio	IPD feed-forward gain
com_u1_flag_volt_err_comp_use	Voltage error compensation setting 0: Disabled 1: Enabled
com_u1_flag_fluxwkn_use	Magnetic flux weakening control setting 0: Disabled 1: Enabled
s_u1_cnt_ics	ICS watchpoint skip counter

Table 5-5 List of variables of the structure for RMW to update parameters

Structure	Variable	Description
st_rmw_param_buffer_t Structure for updating the RMW variables	u2_offset_calc_time	Current offset detection time
	st_motor	Structure for motor parameters
	f4_max_speed_rpm	Maximum speed [rpm]
	u1_ctrl_loop_mode	Control loop mode (position control and speed control)
	u1_encd_angle_adj_mode	Selection of the initial position detection mode
	u2_encd_cpr	Number of pulses per encoder rotation [p/r]
	f4_hs_change_speed_rpm	Switching speed of the speed detection method [rpm]
	f4_hs_change_margin_rpm	Switching speed margin of the speed detection method [rpm]
	f4_ol_ref_id	d-axis current command value in open loop mode [A]
	f4_id_up_time	Setting of the time required for Id increasement
	f4_current_omega_hz	Natural frequency for current control [Hz]
	f4_current_zeta	Attenuation coefficient for current control
	f4_speed_omega_hz	Natural frequency for speed control [Hz]
	f4_speed_zeta	Attenuation coefficient for speed control
	f4_speed_lpf_hz	Speed LPF cut-off frequency [Hz]
	f4_ref_speed_rpm	Speed command value [rpm]
	f4_speed_rate_limit_rpm	Speed variation limit [rpm/s]
	f4_overspeed_limit_rpm	Speed limit value [rpm]
	u1_pos_cmd_mode	Position command status
	u2_pos_interval_time	Position control interval time
	u2_pos_dead_band	Position dead band
	u2_pos_band_limit	Dead band limit value
	f4_pos_omega_hz	Natural frequency for position control [Hz]
	f4_pos_ff_ratio	Position feed-forward gain
	f4_ref_position_deg	Position command value [degrees]
	u1_flag_extobserver_use	Flag for whether to use an observer
	f4_extobs_omega	Natural frequency for speed control module observer [Hz]
	f4_accel_time	Acceleration time [s]
	f4_posprof_max_speed_rpm	Maximum speed value for the position profile (mechanical angl) [rpm]
	u1_flag_ipd_use	Flag for whether to use IPD
	f4_ipd_speed_k_ratio	IPD speed constant

Structure	Variable	Description
st_rmw_param_buffer_t Structure for updating the RMW variables	f4_ipd_pos_kp_ratio	IPD position kp constant
	f4_ipd_omega_hz	IPD control frequency [Hz]
	f4_ipd_pos_ff_ratio	IPD position feed-forward gain
	u1_flag_volt_err_comp_use	Flag for whether to use voltage error compensation
	u1_flag_fluxwkn_use	Flag for whether to use magnetic flux weakening control

5.1.6 Macro definition

Table 5-6 lists macros.

Table 5-6 List of macros

File name	Macro name	Defined value	Remarks
r_app_main.h	MAIN_UI_RMW	0	The RMW UI is used.
	MAIN_UI_BOARD	1	The Board UI is used.
	MAIN_UI_SIZE	2	The number of selectable UIs
r_app_board_ui.h	BOARD_SW1_ON	1	The switch SW1 is on.
	BOARD_SW1_OFF	0	The switch SW1 is off.
	BOARD_SW2_ON	0	The switch SW2 is on.
	BOARD_SW2_OFF	1	The switch SW2 is off.
	BOARD_CHATTERING_CNT	10	The chattering elimination counter value
	BOARD_AD12BIT_DATA	MOTOR_MCU_CFG_AD12BIT_DATA	12-bit AD value
	BOARD_VR1_POSITION_DEAD_BAND	2	Position dead band for VR1 [deg]
	BOARD_VR1_SPEED_DEAD_BAND	80	Speed dead band for VR1 [rpm]
	BOARD_VR1_SPEED_MARGIN	300	Speed margin for VR1 [rpm]
	BOARD_VR1_SCALING_POS	(180 + 18) / (BOARD_AD12BIT_DATA/2 + 1)	Position scaling coefficient for VR1
	BOARD_VR1_SCALING_SPEED	(MOTOR_CFG_MAX_SPEED_RPM + BOARD_VR1_SPEED_MARGIN) / (BOARD_AD12BIT_DATA/2 + 1)	Speed scaling coefficient for VR1
	BOARD_ADJUST_OFFSET	MOTOR_MCU_CFG_ADC_OFFSET	Offset value for VR1
r_app_control_cfg.h	APP_CFG_SCI_CH_SELECT	RX26T: 0x60 RX14T: 0x50	Selection of the SCI channel to be used by users
	APP_CFG_USE_UI	MAIN_UI_RMW	UI initial selection
	APP_CFG_FREQ_BAND_LIMIT	3.0f	Bandwidth limit between control systems [ratio]
	APP_CFG_MAX_CURRENT_OMEGA	1000.0f	Maximum natural frequency for current control [Hz]
	APP_CFG_MIN_OMEGA	1.0f	Minimum natural frequencies [Hz]
r_app_rmw.h	ICS_DECIMATION	RX26T: 5 RX14T: 2	RMW watchpoint skip count
	ICS_INT_LEVEL	6	RMW interrupt priority
	ICS_BRR	RX26T: 251 RX14T: 5	Communication baud rate for RMW
	ICS_INT_MODE	1	Communication mode selection for RMW
	ICS_SCI_CH_SELECT	APP_CFG_SCI_CH_SELECT	SCI channel to be used

5.1.7 Adjustment and configuration of parameters

In the application layer, the configurations must be specified by using the r_app_control_cfg.h file. For the parameters to be set, see 5.1.4.

For the variables listed in Table 5-4, perform adjustment and configuration from RMW. For details about how to use RMW, see 3 Quick Start Guide and the Renesas Motor Workbench User's Manual (R21UZ0004).

5.2 Manager module

The manager module uses specific control modules to control the motor. Its processing includes system-wide management and protection for the interface with each module and for motor control.

5.2.1 Functions

Table 5-7 lists the functions of the manager module.

Table 5-7 List of manager module functions

Function	Description
Mode management	Switches the operation mode of the system in response to the user command to control the motor.
Protection function	Handles errors by using the system protection function.
Control method management	Acquires and sets the states of position control and current control.
Speed and position information acquisition	Acquires the speed and position information.
Control module command value setting	Selects the command values to be entered to the current control module, speed control module, and position control module based on the control states.
Interrupt processing	Assigns processing to appropriate modules in response to interrupts set in Smart Configurator.

5.2.2 Module configuration diagram

Figure 5-6 shows the module configuration.

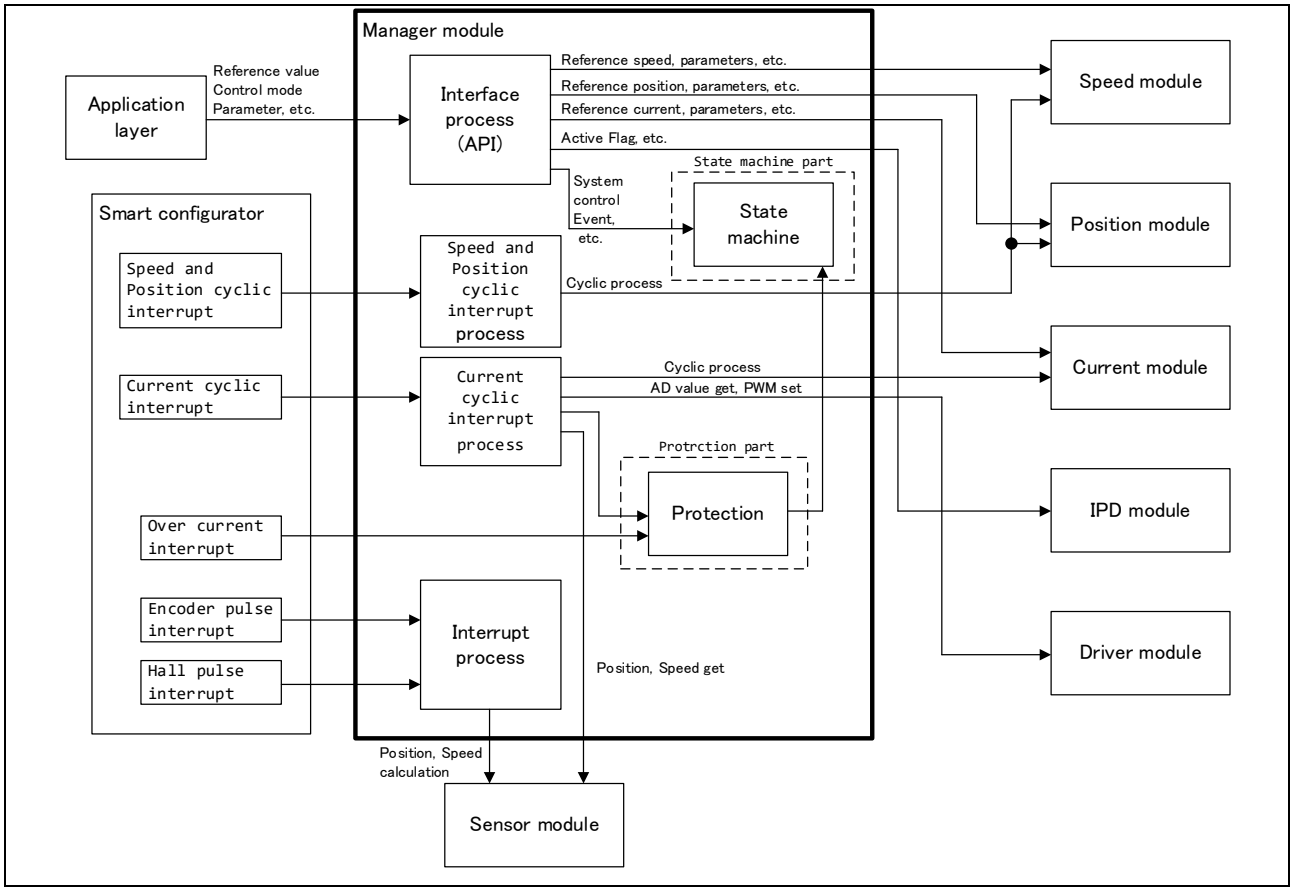


Figure 5-6 Manager module configuration diagram

5.2.3 Mode management

Figure 5-7 shows the state transitions of the target software for this application note. For the target software for this application note, the states are managed by using two types of modes: system modes and run modes. Control Config indicates the control systems that are currently active in the software.

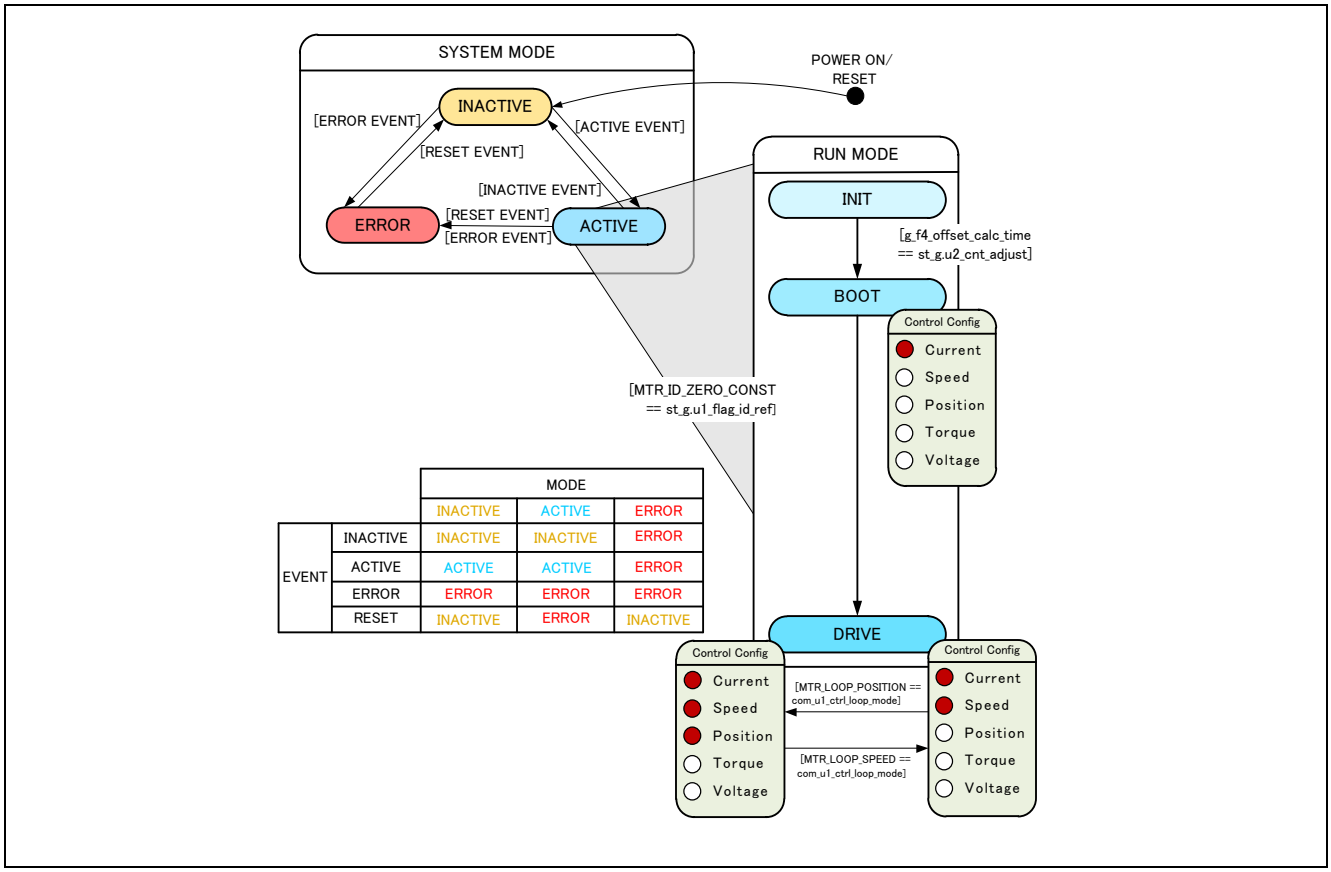


Figure 5-7 State transition diagram for the encoder-based vector control software

(1) System modes

These modes are used to indicate the system operation state. The state transitions as the event corresponding to a new state occurs. There are three system modes: INACTIVE (the motor is stopped), ACTIVE (the motor is running), and ERROR (an error has occurred).

(2) Run modes

These modes are used to indicate the motor control state. When the system enters ACTIVE mode, the motor state transitions as shown in Figure 5-7.

(3) Events

The matrix table in Figure 5-7 shows the system operation state transitions according to the event that occurs in each system mode. The following table shows the trigger that causes each event to occur.

Table 5-8 List of events and their triggers

Event name	Trigger
INACTIVE	Operation performed by the user
ACTIVE	Operation performed by the user
ERROR	Error detection by the system
RESET	Operation performed by the user

5.2.4 Protection function

This control program provides the following error states and implements an emergency stop function in each error state. For details about the values that can be specified for the settings of the system protection function, see Table 5-9.

- Overcurrent error
  Overcurrent errors can be detected on the hardware and in the software.
  A Inactive output is provided to the PWM output pin in response to an emergency stop signal (overcurrent detection) from the hardware.
  This function monitors U-, V-, and W-phases at the overcurrent monitoring interval. When this function detects an overcurrent (the status in which the current is above the overcurrent limit value), it brings the program to an emergency stop (software detection).
  The overcurrent limit value is automatically calculated from the rated current of the motor (MOTOR_CFG_NOMINAL_CURRENT_RMS).
- Overvoltage error
  This function monitors the inverter bus voltage at the overvoltage monitoring interval. When the function detects an overvoltage (the status in which the voltage is above the overvoltage limit value), it brings the program to an emergency stop. The overvoltage limit value is preset in consideration of the conditions such as the error in the resistor value of the detection circuit.
- Low-voltage error
  This function monitors the inverter bus voltage at the low-voltage monitoring interval. When the function detects a low voltage (the status in which the voltage is below the low-voltage limit value), it brings the program to an emergency stop. The low-voltage limit value is preset in consideration of the conditions such as the error in the resistor value of the detection circuit.
- Rotation speed error
  This function monitors the speed at the rotation speed monitoring interval. When the rotation speed exceeds the speed limit value, it brings the program to an emergency stop.

Table 5-9 Values specified for the system protection function settings

Overcurrent error	Overcurrent limit value [A]	2.69
	Monitoring interval [µs]	Current control interval *1
Overvoltage error	Overvoltage limit value [V]	60
	Monitoring interval [µs]	Current control interval *1
Low-voltage error	Low-voltage limit value [V]	8
	Monitoring interval [µs]	Current control interval *1
Rotation speed error	Speed limit value [rpm]	4500
	Monitoring interval [µs]	Current control interval *1

Note: 1. For details, see Table 4-1 Encoder vector control software basic specifications.

5.2.5 Flowcharts

The manager module performs processing in response to the occurrences of interrupts that are set in the smart configurator by using several module API functions to control the motor. The following subsections show the flowcharts of the processing for these interrupts.

5.2.5.1 Interrupt processing for current control

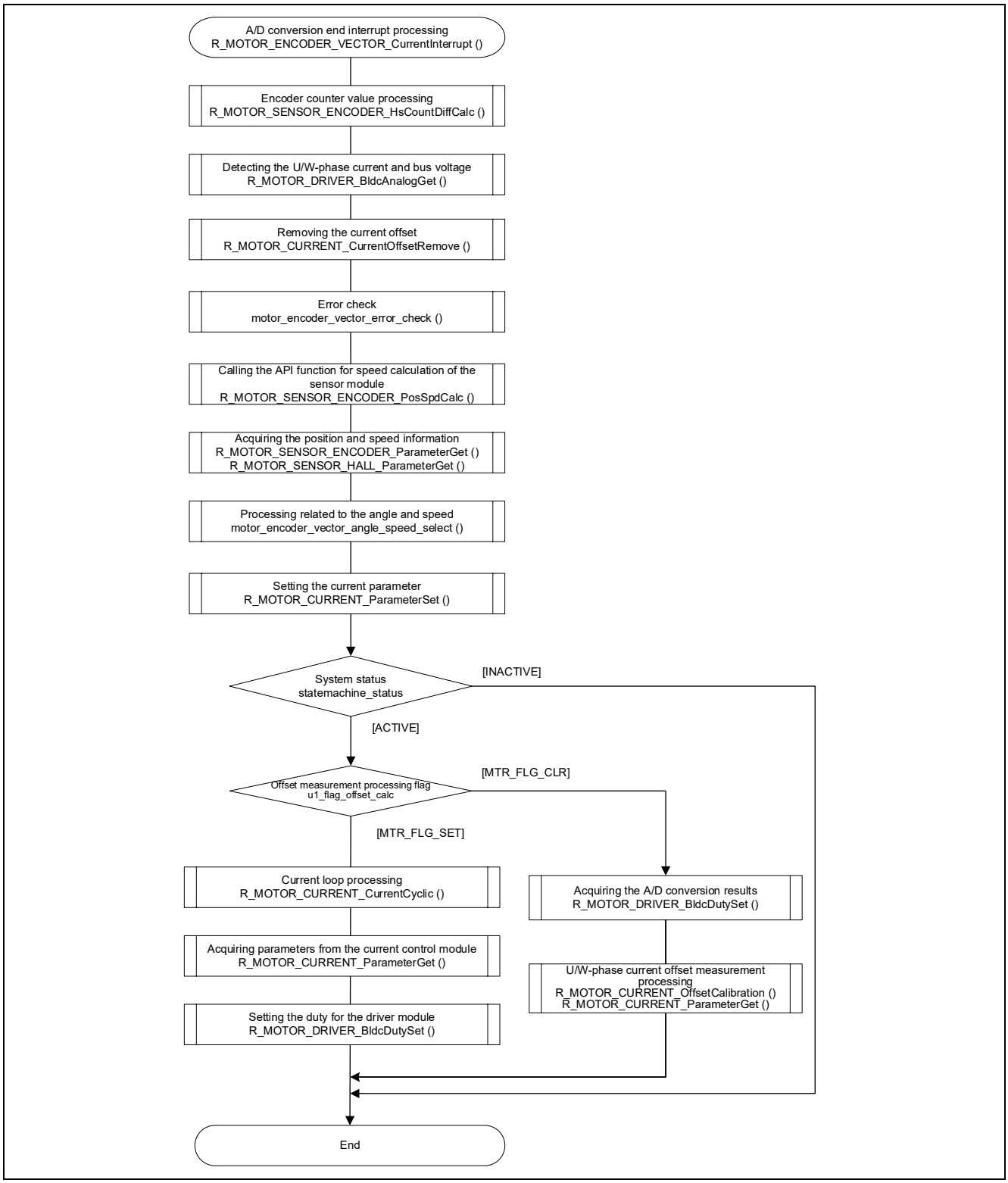


Figure 5-8 Interrupt processing flowchart for current control

5.2.5.2 Interrupt processing for position and speed control

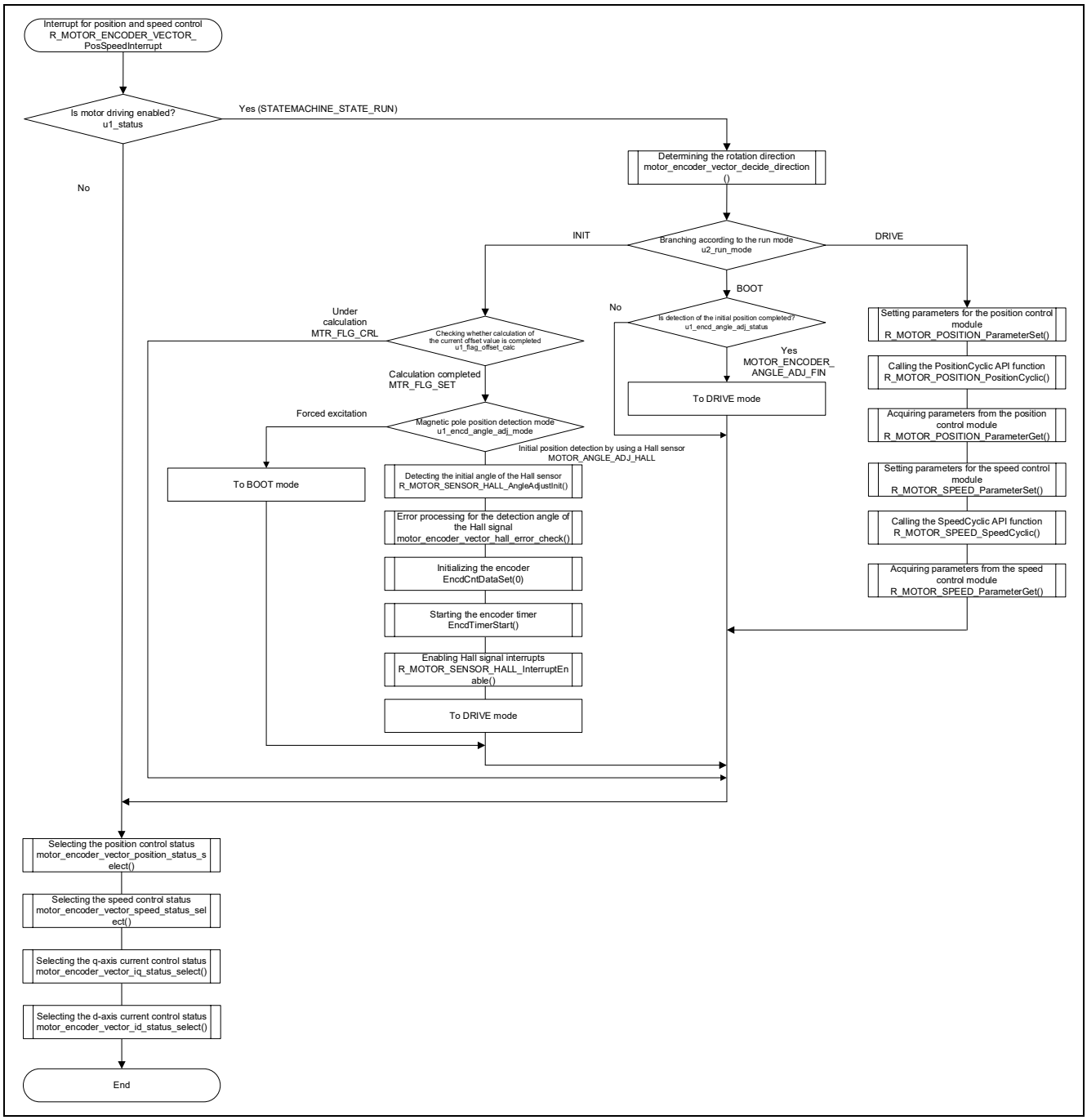


Figure 5-9 Interrupt processing flowchart for position and speed control

5.2.5.3 Overcurrent detection interrupt processing

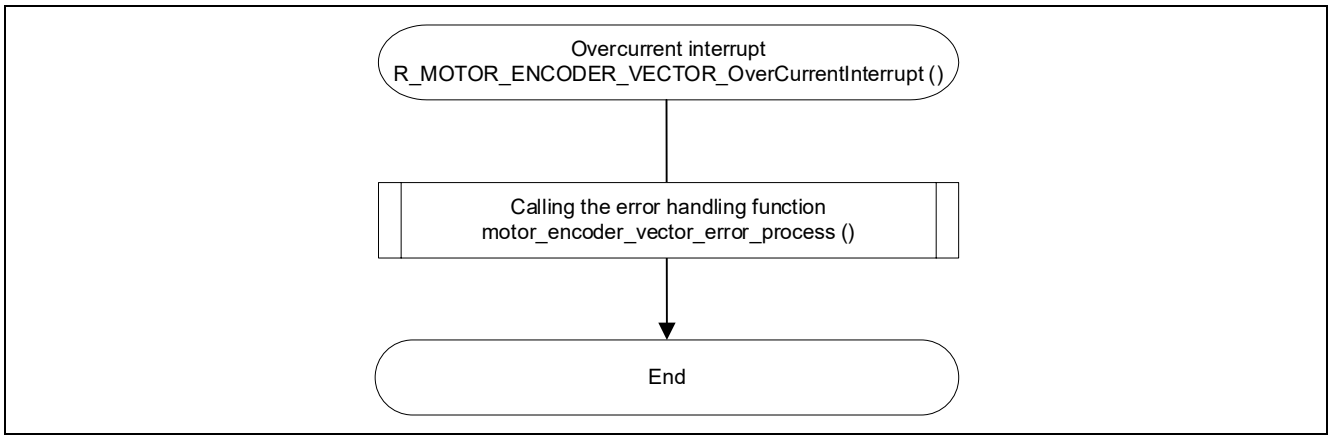


Figure 5-10 Overcurrent detection interrupt processing flowchart

5.2.5.4 Encoder input signal interrupt processing

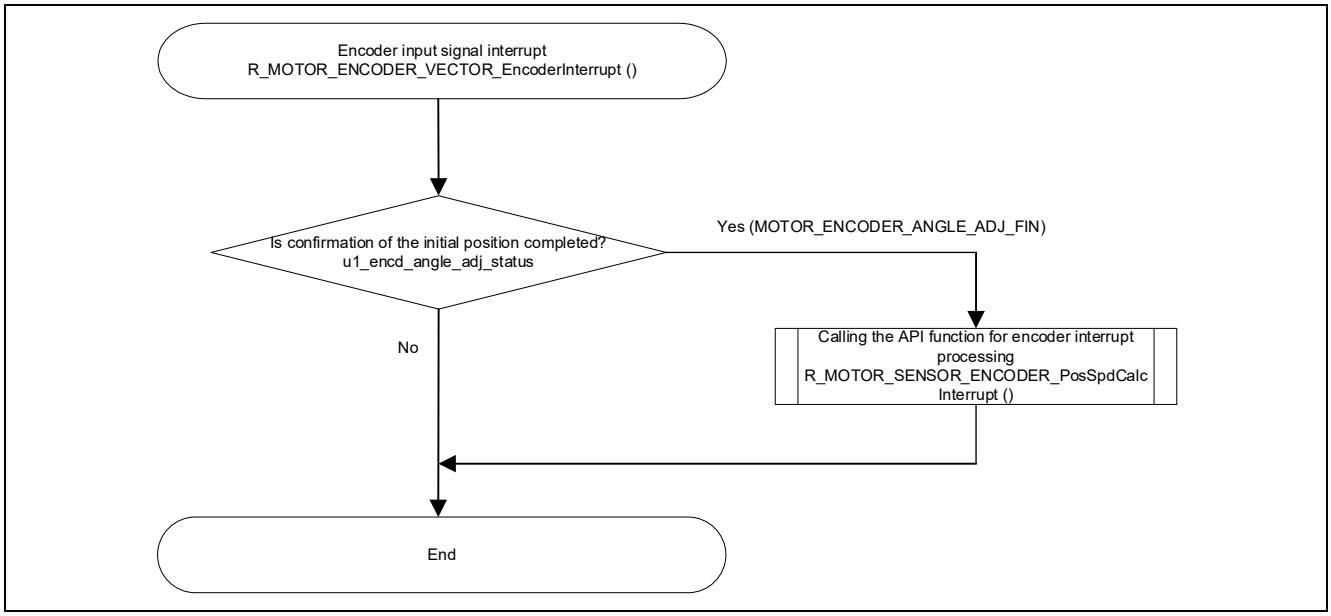


Figure 5-11 Encoder count capturing interrupt processing flowchart

5.2.5.5 Interrupt processing for the Hall signal

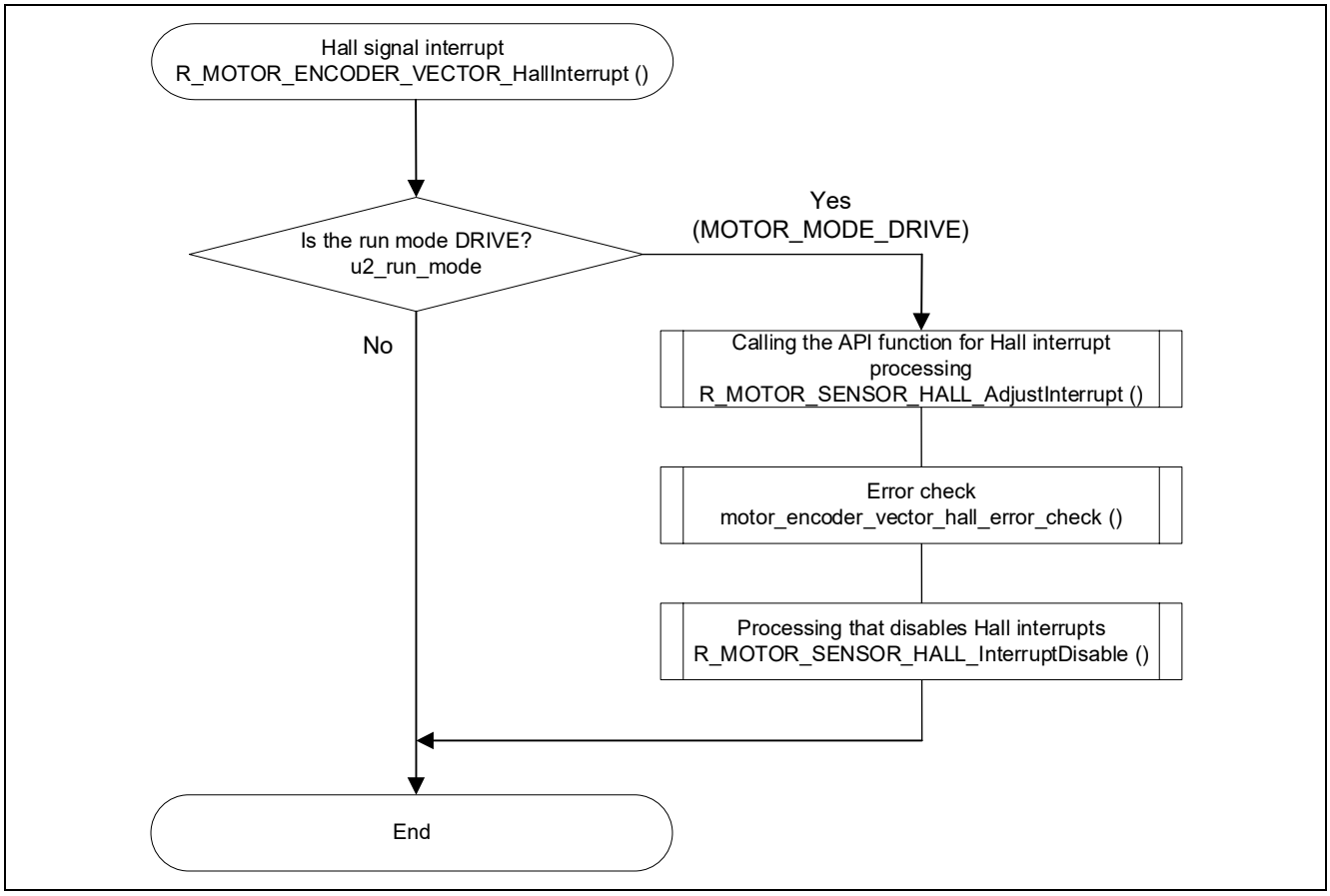


Figure 5-12 Hall edge interrupt processing flowchart

5.2.6 API function

Table 5-10 lists the manager module API functions.

Table 5-10 List of API functions

API function	Description
R_MOTOR_ENCODER_VECTOR_Open	Create an instance of the manager module. Also, execute the Open function of the control module to be used to generate an instance.
R_MOTOR_ENCODER_VECTOR_Close	Reset the manager module and each module.
R_MOTOR_ENCODER_VECTOR_Reset	Initialize the manager module and each module.
R_MOTOR_ENCODER_VECTOR_ParameterUpdate	Updates the control parameter settings of this module. This function also updates the control parameters for the related modules.
R_MOTOR_ENCODER_VECTOR_MotorStart	Places the motor in the running state.
R_MOTOR_ENCODER_VECTOR_MotorStop	Places the motor in the stop state.
R_MOTOR_ENCODER_VECTOR_MotorReset	Releases the system from the error state.
R_MOTOR_ENCODER_VECTOR_ErrorSet	Places the system in an error state.
R_MOTOR_ENCODER_VECTOR_PositionSet	Sets the position command value. This function is enabled when position control is being performed.
R_MOTOR_ENCODER_VECTOR_PositionGet	Acquires the position information.
R_MOTOR_ENCODER_VECTOR_SpeedSet	Sets the speed command value. This function is enabled when speed control is being performed.
R_MOTOR_ENCODER_VECTOR_SpeedGet	Acquires the speed information.
R_MOTOR_ENCODER_VECTOR_StatusGet	Acquires the status from the state machine.
R_MOTOR_ENCODER_VECTOR_ErrorStatusGet	Acquires the error state.
R_MOTOR_ENCODER_VECTOR_CtrlTypeSet	Sets the control method. To change the control method, place the motor in the stop state. 0: Position control 1: Speed control
R_MOTOR_ENCODER_VECTOR_LoopModeStatusGet	Acquires the control method. 0: Position control 1: Speed control
R_MOTOR_ENCODER_VECTOR_PositionCommandModeSet	Allows the user to select the generation mode of position commands for position control. To perform position control, set 1 or 2. 0: The position is always 0. 1: Step response operation 2: Trapezoidal driving operation
R_MOTOR_ENCODER_VECTOR_InPositionFlagGet	Acquires the position control completion state. This function is enabled when position control is being performed. 0: Position control is not completed yet. 1: Position control has been completed.
R_MOTOR_ENCODER_VECTOR_PosSpeedInterrupt	Performs interrupt processing for position control and speed control.
R_MOTOR_ENCODER_VECTOR_CurrentInterrupt	Performs interrupt processing for current control.

API function	Description
R_MOTOR_ENCODER_VECTOR_OverCurrentInterrupt	Performs interrupt processing when an overcurrent occurs.
R_MOTOR_ENCODER_VECTOR_HallInterrupt	Performs interrupt processing for Hall sensor signals.
R_MOTOR_ENCODER_VECTOR_EncoderInterrupt	Performs interrupt processing for encoder signals.
R_MOTOR_ENCODER_VECTOR_IPDEnableSet	Enables or disables the IPD module. 0: Disable 1: Enable

5.2.7 Configurations

Table 5-11 lists the configurations for the manager module.

Table 5-11 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	MOTOR_MCU_CFG_PWM_TIMER_FREQ	PWM timer frequency [MHz]
	MOTOR_MCU_CFG_CARRIER_FREQ	Carrier wave frequency [kHz]
	MOTOR_MCU_CFG_INTR_DECIMATION	Skipping count for carrier wave interrupts
	MOTOR_MCU_CFG_AD_FREQ	AD operating frequency [MHz]
	MOTOR_MCU_CFG_AD_SAMPLING_CYCLE	AD sampling state [cycles]
	MOTOR_MCU_CFG_AD12BIT_DATA	AD resolution
	MOTOR_MCU_CFG_ADC_OFFSET	AD intermediate data
	MOTOR_TYPE_BLDC	Enable this item to use the BLDC motor.
	MOTOR_COMMON_CFG_LOOP_MODE	Sets the default run mode.
	MOTOR_COMMON_CFG_IPD_CTRL	Sets whether to use the default IPD.
	MOTOR_COMMON_CFG_OVERCURRENT_MARGIN_MULT	Overcurrent limit value [A]
	MOTOR_COMMON_CFG_IA_MAX_CALC_MULT	Coefficient for calculating the overcurrent limit value BLDC: Square root of 3 STM: Square root of 2
	MOTOR_MCU_CFG_TFU_OPTIMIZE	Sets the processing of TFU-specific functions MTR_ENABLE MTR_DISABLE

Table 5-12 List of initial values for configurations

Macro name	Settings
MOTOR_MCU_CFG_PWM_TIMER_FREQ	CG_CONFIG_MOTOR_PWM_TIMER_FREQ
MOTOR_MCU_CFG_CARRIER_FREQ	CG_CONFIG_MOTOR_CARRIER_FREQ
MOTOR_MCU_CFG_INTR_DECIMATION	RX26T: CG_CONFIG_MOTOR_INTR_DECIMATION RX14T: 1
MOTOR_MCU_CFG_AD_FREQ	CG_MOTOR_MCU_CFG_AD_FREQ
MOTOR_MCU_CFG_AD_SAMPLING_CYCLE	RX26T: 45 RX14T: 10.5
MOTOR_MCU_CFG_AD12BIT_DATA	CG_MOTOR_CFG_MAX_AD_DATA
MOTOR_MCU_CFG_ADC_OFFSET	0x7FF
MOTOR_TYPE_BLDC	With definition
MOTOR_COMMON_CFG_LOOP_MODE	MOTOR_LOOP_SPEED
MOTOR_COMMON_CFG_IPD_CTRL	MTR_DISABLE
MOTOR_COMMON_CFG_OVERCURRENT_MARGIN_MULT	1.5
MOTOR_COMMON_CFG_IA_MAX_CALC_MULT	MTR_SQRT_3
MOTOR_MCU_CFG_TFU_OPTIMIZE	MTR_ENABLE

5.2.8 Structure and variable information

Table 5-13 lists the structures and variables for the manager module. For the manager module, the structure for the manager module (g_st_encoder_vector) is defined by securing an instance of the module from the API.

Table 5-13 List of structures and variables

Structure	Variable	Description
st_encoder_vector_control_t Structure for the manager module	u1_direction	Rotation direction 0：CW 1：CCW
	u1_ctrl_loop_mode	Control mode selection 0: Position control 1: Speed control
	u1_encd_angle_adj_mode	Selection of the initial position detection mode 0: Forced excitation 1: Use of hall sensor
	u1_encd_angle_adj_status	Status of detection of the initial position
	u2_encd_angle_adj_time	Wait time for detection of the initial position [µs]
	u2_encd_angle_adj_cnt	Counter for the wait time for detection of the initial position
	u1_flag_ipd_use	Flag for whether to use IPD 0: IPD control disabled 1: IPD control enabled
	u2_error_status	Error status
	u2_run_mode	Run mode 0: Initialization 1: Preparation for start 2: Motor drive
	u1_state_id_ref	Status of d-axis current command value
	u1_state_iq_ref	Status of q-axis current command value
	u1_state_speed_ref	Status of speed command value
	u2_encd_cpr	Number of pulses per encoder rotation
	f4_vdc_ad	Bus voltage [V]
	f4_iu_ad	U-phase current [A]
	f4_iv_ad	V-phase current [A]
	f4_iw_ad	W-phase current [A]
	f4_overcurrent_limit	Overcurrent limit value [A]
	f4_overvoltage_limit	Overvoltage limit value [V]
	f4_undervoltage_limit	Low-voltage limit value [V]
	f4_overspeed_limit_rad	Overspeed limit value [rad/s]
	f4_max_speed_rc_fil_rpm	Maximum speed command value of rc filter [rpm] (mechanical)
	f4_rotor_angle_rad	Rotor angle [rad]

Structure	Variable	Description
st_encoder_vector_control_t Structure for the manager module	st_current_output	Structure for current control module output
	st_speed_output	Structure for speed control module output
	st_position_output	Structure for position control module output
	st_sensor_encoder_output	Structure for encoder output of the sensor module
	st_sensor_hall_output	Structure for Hall output of the sensor module
	st_stm	Structure for the state machine
	st_motor	Motor parameter structure
	*p_st_cc	Current control module generation instance pointer
	*p_st_sc	Speed control module generation instance pointer
	*p_st_pc	Position control module generation instance pointer
	*p_st_ipd	IPD control module generation instance pointer
	*p_st_sensor	Sensor module generation instance pointer
	*p_st_driver	Driver module generation instance pointer
st_encoder_vector_cfg_t Structure for setting the manager module control parameters	u1_encd_angle_adj_mode	Selection of the initial position detection mode
	u2_encd_cpr	Number of pulses per encoder rotation
	f4_overspeed_limit_rpm	Speed limit value [rpm]
	st_motor	Motor parameter structure

5.2.9 Macro definition

Table 5-14 lists the macros for the manager module.

Table 5-14 List of macros

File name	Macro name	Defined value	Remarks
r_motor_encoder_vector_api.h	MOTOR_LOOP_POSITION	0	Position control mode
	MOTOR_LOOP_SPEED	1	Speed control mode
	MOTOR_ENCODER_VECTOR_ERROR_NONE	(0x0000)	An error status. There is no error.
	MOTOR_ENCODER_VECTOR_ERROR_OVER_CURRENT_HW	(0x0001)	An error status. A hardware overcurrent error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_OVER_VOLTAGE	(0x0002)	An error status. An overvoltage error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_OVER_SPEED	(0x0004)	An error status. An overspeed error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_HALL_PATTERN	(0x0020)	An error status. A Hall pattern error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_LOW_VOLTAGE	(0x0080)	An error status. A low-voltage error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_OVER_CURRENT_SW	(0x0100)	An error status. A software overcurrent error has occurred.
	MOTOR_ENCODER_VECTOR_ERROR_UNKNOWN	(0xffff)	An error status. An error whose error code is unknown has occurred.
	MOTOR_ANGLE_ADJ_EXCIT	0	Initial position detection by forced excitation
	MOTOR_ANGLE_ADJ_HALL	1	Initial position detection by using the Hall sensor
	MOTOR_CTRL_TYPE_POS	0	Macro for switching control methods. Position control mode.
	MOTOR_CTRL_TYPE_SPEED	1	Macro for switching control methods. Speed control mode.
r_motor_encoder_vector_manager.h	MOTOR_MODE_INIT	(0x00)	INIT run mode
	MOTOR_MODE_BOOT	(0x01)	BOOT run mode
	MOTOR_MODE_DRIVE	(0x02)	DRIVE run mode
	MOTOR_ENCODER_ANGLE_ADJ_90DEG	0	Status at startup. The encoder is positioned at 90 degrees.
	MOTOR_ENCODER_ANGLE_ADJ_0DEG	1	Status at startup. The encoder is positioned at 0 degrees.
	MOTOR_ENCODER_ANGLE_ADJ_FIN	2	Status at startup. Determination of the initial position is completed.

5.2.10 Adjustment and configuration of parameters

When you use the sample program, you need to correctly set the inverter information and the information about the motor to be used. Table 5-15 shows the values set in the sample program.

Table 5-15 Motor and inverter parameter settings

File name	Macro name	Settings	Description
r_motor_inverter_cfg.h	INVERTER_CFG_SHUNT_RESIST	0.010f	Shunt resistance value [ohm]
	INVERTER_CFG_DEADTIME	CG_CONFIG_MOTOR_DEADTIME	Dead time [µs]
	INVERTER_CFG_VOLTAGE_GAIN	22.2766f	Coefficient for voltage detection
	INVERTER_CFG_CURRENT_AMP_GAIN	20.0f	Gain of the amplifier for current detection
	INVERTER_CFG_CURRENT_LIMIT	21.4f	Overcurrent limit value for the inverter board [A]
	INVERTER_CFG_OVERVOLTAGE_LIMIT	60.0F	Overvoltage limit [V]
	INVERTER_CFG_UNDERVOLTAGE_LIMIT	8.0f	Low-voltage limit [V]
	INVERTER_CFG_INPUT_V	24.0f	Input voltage [V]
	INVERTER_CFG_ADC_REF_VOLTAGE	5.0f	Analog power supply voltage for the MCU [V]
	INVERTER_CFG_COMP_V0	RX26T: 0.564f RX14T: 0.234f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V1	RX26T: 0.782f RX14T: 0.391f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V2	RX26T: 0.937f RX14T: 0.463f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V3	RX26T: 1.027f RX14T: 0.508f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V4	RX26T: 1.058f RX14T: 0.524f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_I0	RX26T: 0.022f RX14T: 0.035f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I1	RX26T: 0.038f RX14T: 0.066f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I2	RX26T: 0.088f RX14T: 0.131f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I3	RX26T: 0.248f RX14T: 0.313f	Coefficient for compensation of the voltage error [A] *1

File name	Macro name	Settings	Description
r_motor_inverter_cfg.h	INVERTER_CFG_COMP_I4	RX26T: 0.865f RX14T: 0.609f	Coefficient for compensation of the voltage error [A] *1
r_motor_targetmotor_cfg.h	MOTOR_CFG_POLE_PAIRS	4	Number of pole pairs
	MOTOR_CFG_MAGNETIC_FLUX	0.006612919f	Magnetic flux [wb]
	MOTOR_CFG_RESISTANCE	0.8933714f	Resistance [ohm]
	MOTOR_CFG_D_INDUCTANCE	0.001091948f	d-axis inductance [H]
	MOTOR_CFG_Q_INDUCTANCE	0.001091948f	q-axis inductance [H]
	MOTOR_CFG_ROTOR_INERTIA	0.000002647f	Rotor inertia [kg m^2]
	MOTOR_CFG_NOMINAL_CURRENT_RMS	1.27f	Rated current [A]
	MOTOR_CFG_MAX_SPEED_RPM	4000.0f	Maximum speed [rpm]

Note: 1. For details, see 5.5 Voltage error compensation (current control module).

5.2.11 Startup sequence management

The manager module controls the motor by changing the flag settings that manage the command values for the d-axis current, q-axis current, speed, and position according to the run mode. Also, by changing these command values appropriately, the manager module creates a starting sequence to start the motor. Figure 5-13 and Figure 5-14 show motor startup control based on encoder-based vector control. Note that the motor can be started by two methods: forced excitation (using only the encoder) and by using a Hall sensor. This section describes these two startup methods.

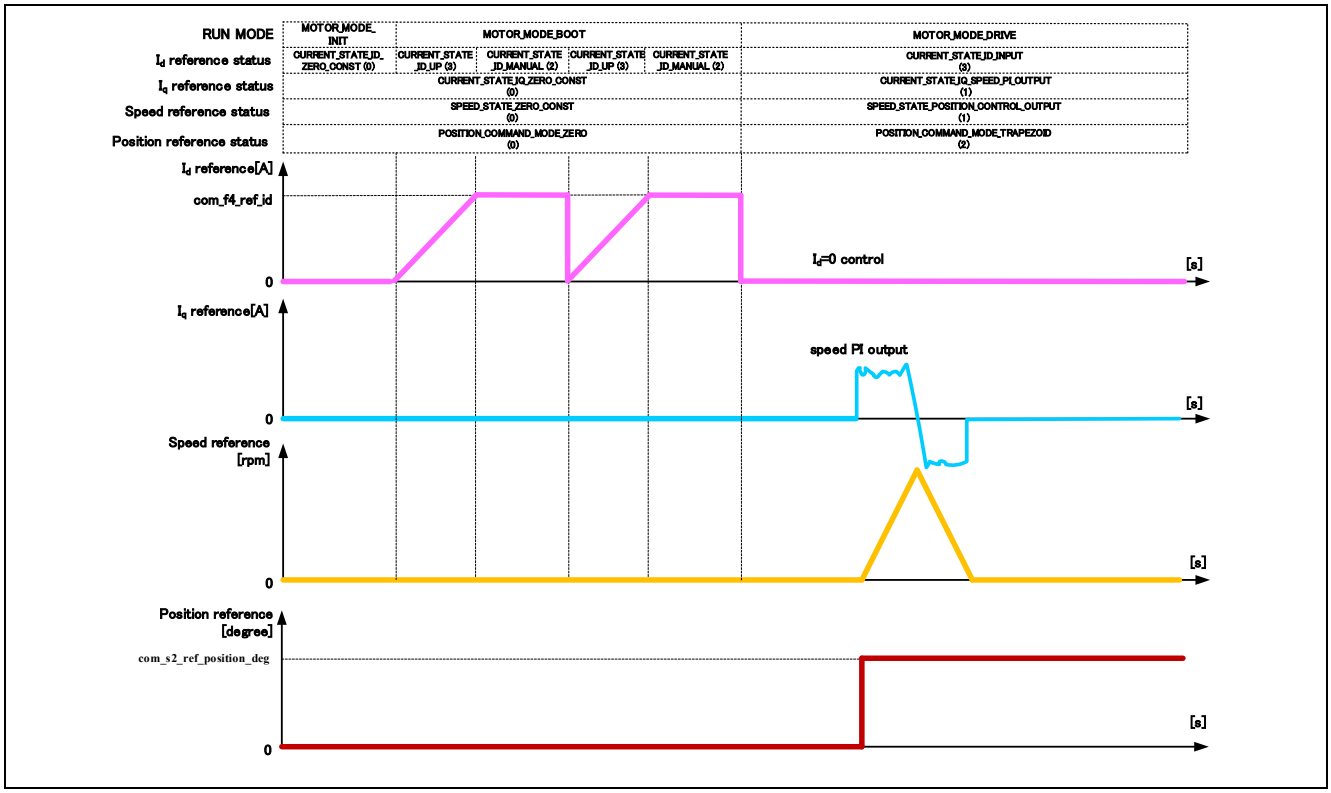


Figure 5-13 Motor start control based on encoder-based vector control (in the case of position control)

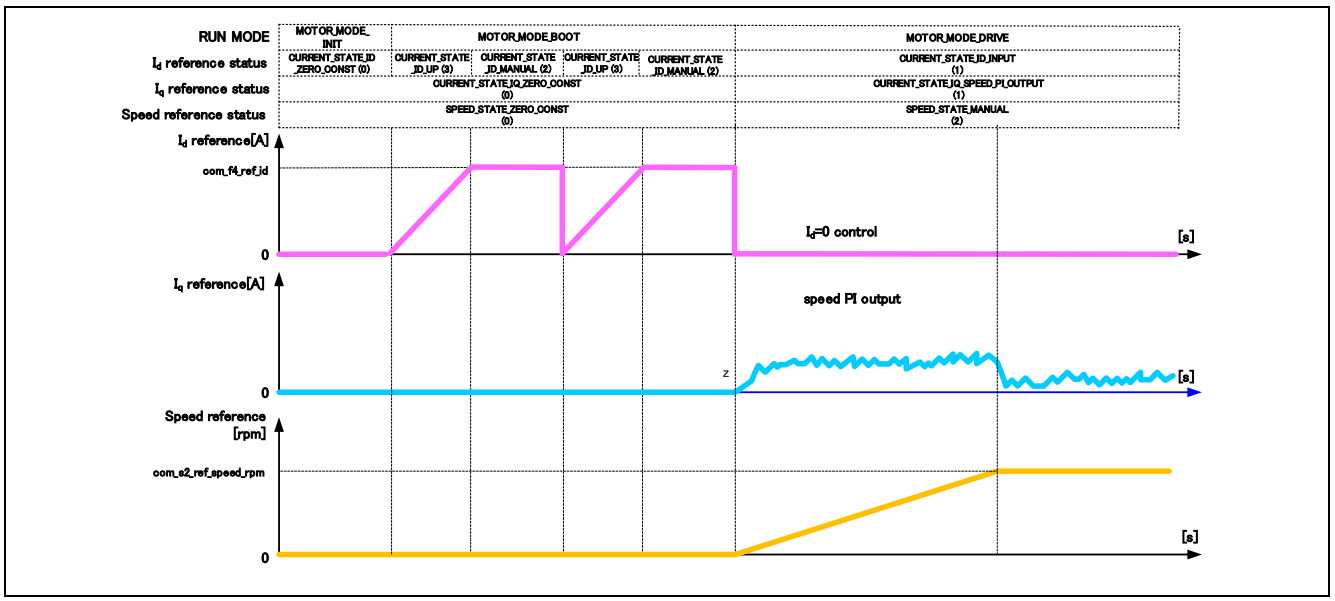


Figure 5-14 Motor start control based on encoder-based vector control (in the case of speed control)

(1) Determining the magnetic pole position by using only an encoder

If you use an incremental encoder as the position sensor, you can only obtain the relative position information (not the absolute magnetic pole position information). In this case, therefore, the initial magnetic pole position must have been determined at startup of the motor. This system determines the initial magnetic pole position by creating the current vector in the sequence shown in Figure 5-15 to pull in the magnet so that the directions of the d axis and current vector match. Figure 5-16 shows the startup sequence that applies in this case.

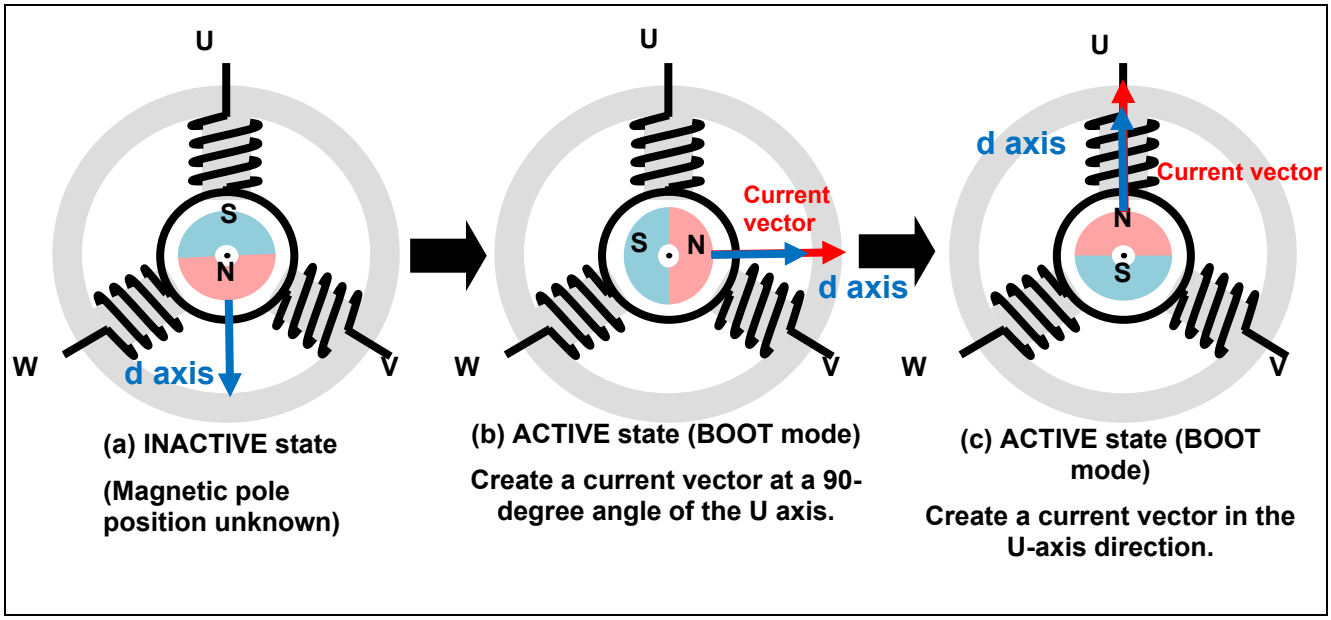


Figure 5-15 Determining the position of a permanent magnet

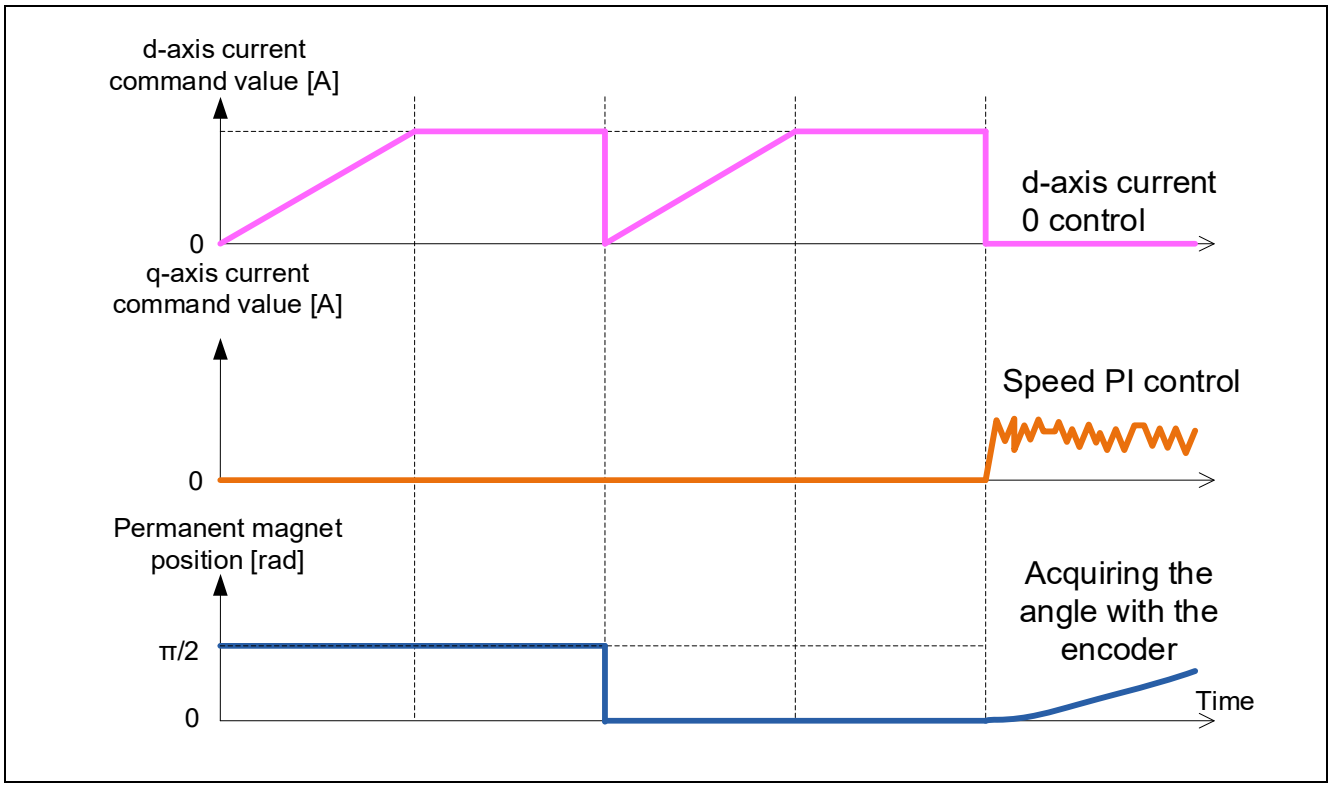


Figure 5-16 Startup sequence based on encoder-based vector control (example)

(2) Magnetic pole position detection by using a Hall sensor

Use of a Hall sensor together with an encoder allows for detection of the magnetic pole position. Figure 5-17 shows how the magnetic pole position is determined in the case where a Hall sensor is used.

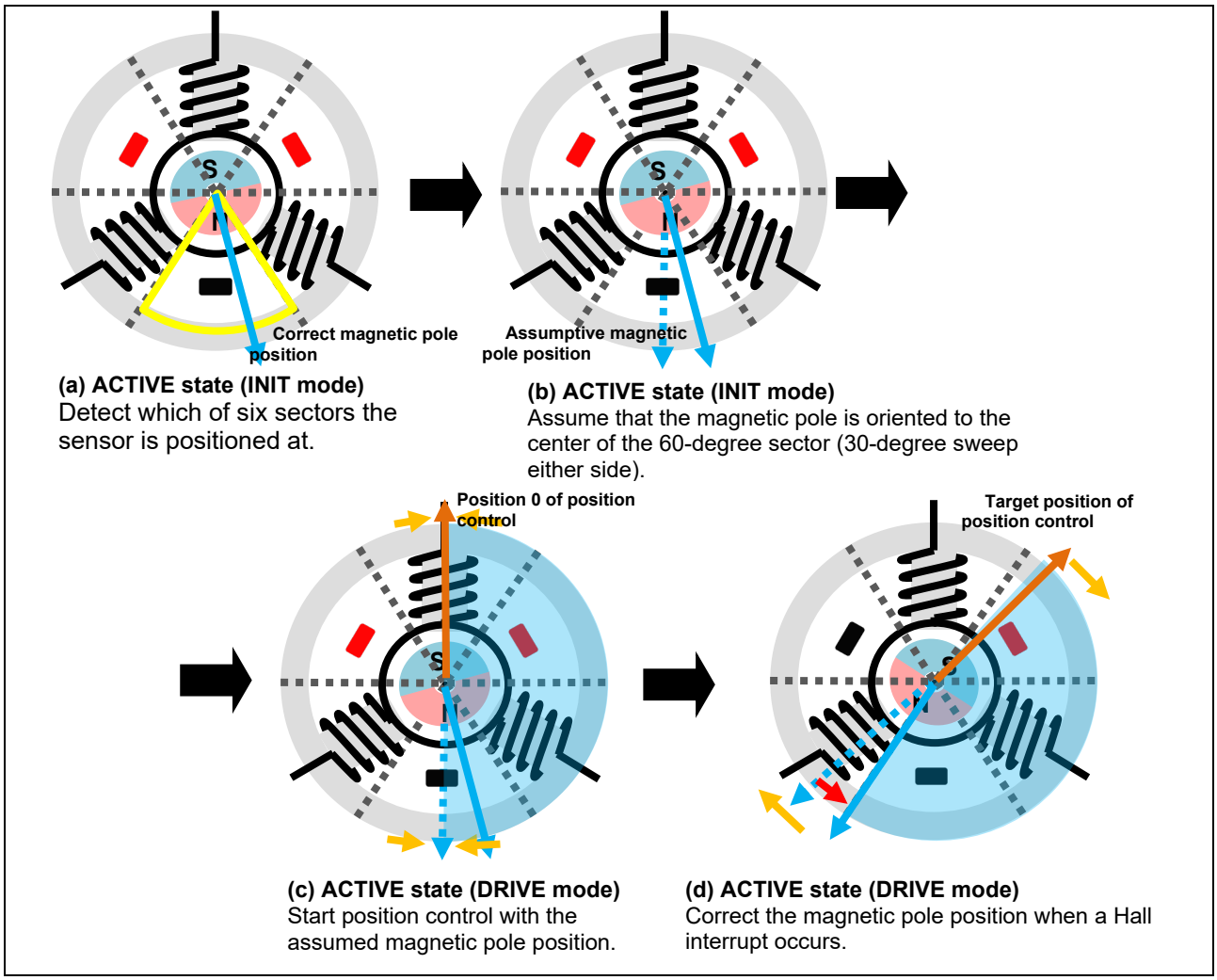


Figure 5-17 Procedure for determining the magnetic pole position of a permanent magnet

(a) The system uses a Hall sensor to detect which of six 60-degree sectors the sensor is positioned at.
(b) The system uses the central orientation of the detected 60-degree sector (with a 30-degree sweep either side) as the initial magnetic pole position.
(c) The system performs position control until a Hall interrupt occurs (up to 30 degrees sweep either side of the pole position).
(d) Upon detecting a Hall edge, the system adjusts the vector-controlled angle to the correct magnetic pole position.
With additional use of a Hall sensor, the system can proceed to positioning control without performing pull-in operation.

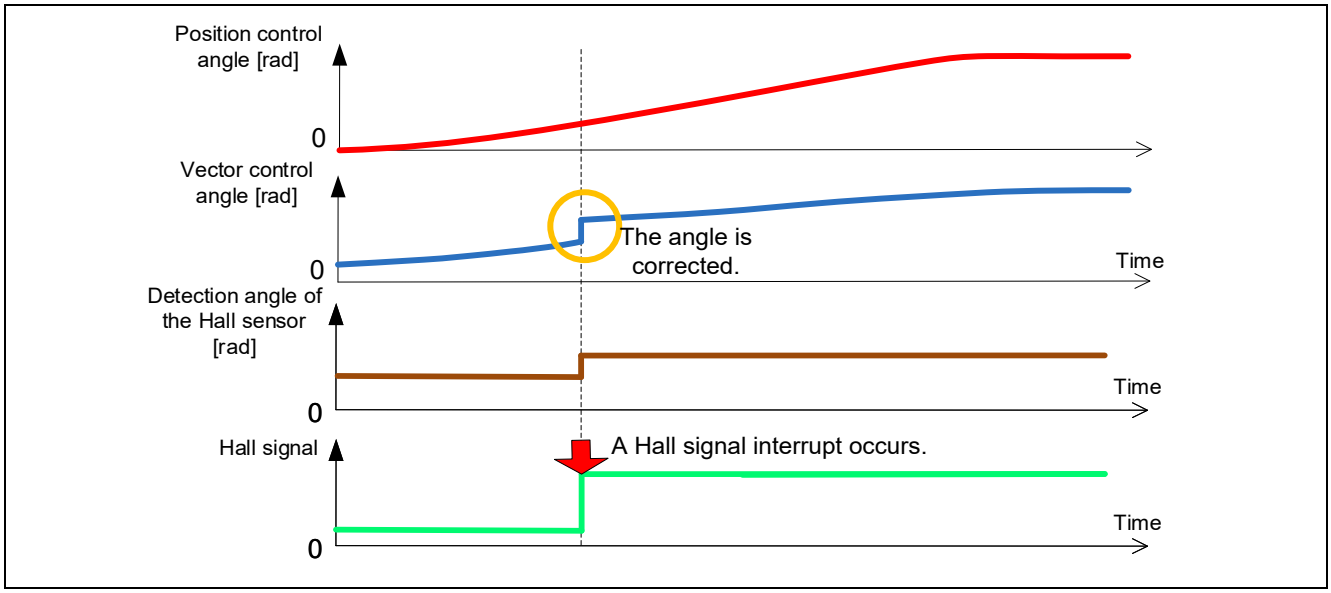


Figure 5-18 Startup sequence in initial position detection using a Hall sensor

5.3 Current control module

The current control module uses the value of the incoming current to perform coordinate transformation and feedback control that are necessary for vector control, and then calculates the voltage of the PWM output. The module also controls submodules that perform modulation and voltage error compensation.

5.3.1 Function

Table 5-16 lists the functions of the current control module.

Table 5-16 List of functions of the current control module

Function	Description
Current control	Performs calculation according to the current command value to set the PWM output value.
Current offset adjustment	Calculates the offset value of the current value detected by AD.
Voltage error compensation	Compensates for the effects of output voltage dead time.
Coordinate transformation and inverse transformation	Performs coordinate transformation for the current value detected to perform vector control. This function also performs inverse transformation of the coordinate for the calculation results to restore the original coordinate axis.
Modulation	Improves the efficiency by modulation to a PWM signal.
Non-interacting control	Calculates interference cancellation to prevent interference between the d and q axes.

5.3.2 Module configuration diagram

Figure 5-19 shows the module configuration.

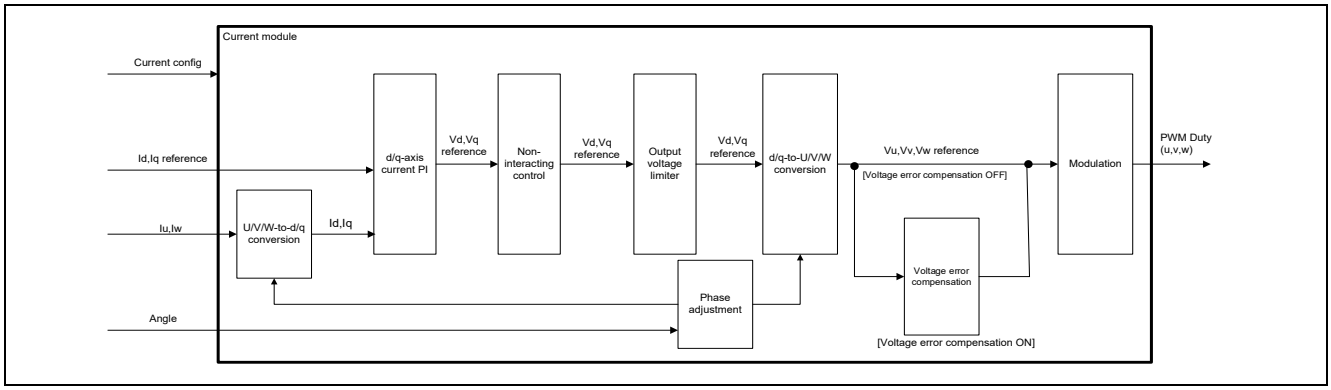


Figure 5-19 Current control module configuration diagram

5.3.3 Flowchart

Figure 5-20 shows the flowchart for the loop processing of the current control module.

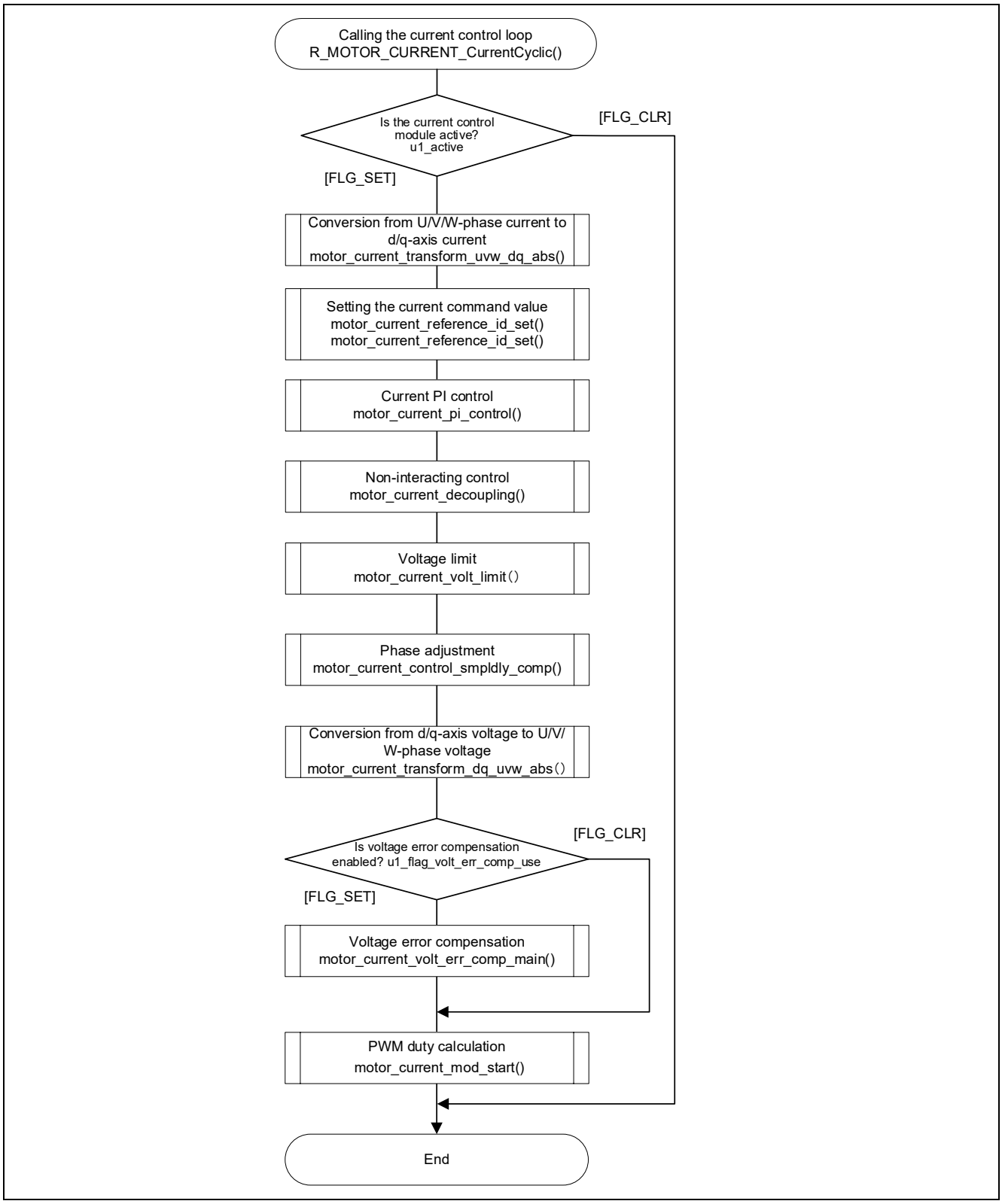


Figure 5-20 Flowchart for the loop processing of the current control module

5.3.4 API function

Table 5-17 lists the API functions of the current control module.

Table 5-17 List of API functions

API function	Description
R_MOTOR_CURRENT_Open	Generates an instance of the current control module.
R_MOTOR_CURRENT_Close	Places the current control module in a reset state.
R_MOTOR_CURRENT_Reset	Initializes the current control module.
R_MOTOR_CURRENT_Run	Activates the current control module.
R_MOTOR_CURRENT_ParameterSet	Inputs the variable information that is used for current control.
R_MOTOR_CURRENT_ParameterGet	Acquires the current control results that are output.
R_MOTOR_CURRENT_ParameterUpdate	Updates the control parameters of the current control module.
R_MOTOR_CURRENT_CurrentCyclic	Performs current control.
R_MOTOR_CURRENT_OffsetCalibration	Performs offset adjustment of current detection.
R_MOTOR_CURRENT_OffsetRemove	Returns the value with the current detection offset value excluded.
R_MOTOR_CURRENT_VoltErrCompParamSet	Sets the voltage error compensation parameter.

5.3.5 Configurations

Table 5-18 shows the configurations that are used for the current control module. Set up the functions to be used and the necessary parameters. Table 5-19 shows the initial values.

Table 5-18 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	CURRENT_CFG_VOLT_ERR_COMP	Enables or disables the voltage error compensation function. Enable: MTR_ENABLE Disable: MTR_DISABLE
	CURRENT_CFG_MODULATION_METHOD	Modulation method MOD_METHOD_SPWM: Sinusoidal PWM MOD_METHOD_SVPWM: Spatial vector PWM
	CURRENT_CFG_OFFSET_CALC_TIME	Current offset measurement time [ms]
	CURRENT_CFG_PERIOD_MAG_VALUE	Coordinate transformation interval coefficient
	CURRENT_CFG_PI_INTEGRAL_LIMIT_VD	d-axis current limit [V] INVERTER_CFG_INPUT_V: The maximum input voltage is defined by using r_motor_inverter_cfg.h.
	CURRENT_CFG_PI_INTEGRAL_LIMIT_VQ	q-axis current limit [V]
	CURRENT_CFG_OMEGA	Natural frequency for current control [Hz]
	CURRENT_CFG_ZETA	Attenuation coefficient for current control
	CURRENT_CFG_REF_ID_OPENLOOP	d-axis current command value in open loop mode [A]
	CURRENT_CFG_ID_UP_STEP_TIME	Additional time for the d-axis current command value

Table 5-19 List of initial values for configurations

Macro name	Settings
CURRENT_CFG_VOLT_ERR_COMP	MTR_ENABLE
CURRENT_CFG_MODULATION_METHOD	MOD_METHOD_SVPWM
CURRENT_CFG_OFFSET_CALC_TIME	RX26T: 512.0f RX14T: 256.0f
CURRENT_CFG_PERIOD_MAG_VALUE	1.0f
CURRENT_CFG_PI_INTEGRAL_LIMIT_VD	INVERTER_CFG_INPUT_V * 0.5f
CURRENT_CFG_PI_INTEGRAL_LIMIT_VQ	INVERTER_CFG_INPUT_V * 0.5f
CURRENT_CFG_OMEGA	300.0f
CURRENT_CFG_ZETA	1.0f
CURRENT_CFG_REF_ID_OPENLOOP	1.5f
CURRENT_CFG_ID_UP_STEP_TIME	RX26T: 2560.0f RX14T: 1280.0f

5.3.6 Structure and variable information

Table 5-20 lists the structures and variables that are used for the current control module. For the current control module, the structure for the current control module (g_st_cc) is defined by securing an instance of the module from the API.

Table 5-20 List of structures and variables

Structure	Variable	Description
st_current_control_t Structure for the current control module	u1_active	The active state of the current control module
	u1_flag_volt_err_comp_use	Enables or disables the voltage error compensation function.
	u1_state_id_ref	Status of the d axis at startup
	u1_state_iq_ref	Status of the q axis at startup
	u1_flag_offset_calc	Flag for the current offset calculation
	u2_offset_calc_time	Measurement time in current offset adjustment [ms]
	u2_crnt_offset_cnt	Measurement count in current offset adjustment [ms]
	f4_ctrl_period	Current control interval (period) [s]
	f4_refu	U-phase command voltage [V]
	f4_refv	V-phase command voltage [V]
	f4_refw	W-phase command voltage [V]
	f4_vd_ref	d-axis voltage command value [V]
	f4_vq_ref	q-axis voltage command value [V]
	f4_id_ref	d-axis current command value [A]
	f4_iq_ref	q-axis current command value [A]
	f4_id_ad	d-axis current value [A]
	f4_iq_ad	q-axis current value [A]
	f4_lim_iq	q-axis current limit value [A]
	f4_offset_iu	U-phase offset current value [A]
	f4_offset_iw	W-phase offset current value [A]
	f4_sum_iu_ad	U-phase total current value [A]
	f4_sum_iw_ad	W-phase total current value [A]
	f4_vdc_ad	Bus voltage value [V]
	f4_iu_ad	U-phase current value [A]
	f4_iv_ad	V-phase current value [A]
	f4_iw_ad	W-phase current value [A]
	f4_modu	U-phase duty cycle

Structure	Variable	Description
st_current_control_t Structure for the current control module	f4_modv	V-phase duty cycle
	f4_modw	W-phase duty cycle
	f4_speed_rad	Speed [rad/s]
	f4_rotor_angle_input_rad	Rotor angle [rad]
	f4_id_up_step	Variation when the ID is set [A]
	f4_ref_id_ctrl	The reference d-axis current value [A]
	f4_ref_iq_ctrl	The reference q-axis current value [A]
	f4_ol_ref_id	d-axis current command value in open loop mode [A]
	f4_va_max	Maximum voltage on the d and q axes [V]
	f4_id_ref_buff	Buffer size for the d-axis current command value [A]
	st_mod	Structure for modulation
	st_volt_comp	Structure for voltage error compensation
	st_pi_id	Structure for d-axis PI control
	st_pi_iq	Structure for q-axis PI control
	st_rotor_angle_t	Structure for rotor information
	st_motor	Structure for motor parameters
st_current_cfg_t Structure for setting the parameters for controlling the current control module	u2_offset_calc_time	Offset calculation time
	f4_ctrl_period	Control interval [s]
	f4_current_omega_hz	Natural frequency for current control [Hz]
	f4_current_zeta	Attenuation coefficient for current control
	u1_flag_volt_err_comp_use	Enables or disables voltage error compensation.
	f4_id_up_step	Increment of the d-axis current
	f4_ol_ref_id	d-axis current command value in open loop mode [A]
	st_motor	Structure for motor parameters
st_current_output_t Structure for the current control module output	u1_flag_offset_calc	Current offset flag
	f4_modu	U-phase duty cycle
	f4_modv	V-phase duty cycle
	f4_modw	W-phase duty cycle
	f4_neutral_duty	Duty cycle in offset measurement
	f4_va_max	Maximum voltage on the d and q axes [V]

Structure	Variable	Description
st_current_input_t Structure for the current control module input	u1_state_id_ref	Status of the d axis
	u1_state_iq_ref	Status of the q axis
	f4_rotor_angle_rad	Rotor angle [rad]
	f4_iu_ad	U-phase current value [A]
	f4_iv_ad	V-phase current value [A]
	f4_iw_ad	W-phase current value [A]
	f4_vdc_ad	Bus voltage value [V]
	f4_speed_rad	Speed [rad/s]
	f4_id_ref	d-axis current command value [A]
	f4_iq_ref	q-axis current command value [A]

5.3.7 Macro definition

Table 5-21 lists the macros that are used for the current control module.

Table 5-21 List of macros

File name	Macro name	Defined value	Description
r_motor_current_api.h	CURRENT_STATE_ID_ZERO_CONST	0	Current status for the d axis: d-axis current always-0 mode
	CURRENT_STATE_ID_INPUT	1	Current status for the d axis: d-axis current command input mode
	CURRENT_STATE_ID_MANUAL	2	Current status for the d axis: d-axis fixed-command mode
	CURRENT_STATE_ID_UP	3	Current status for the d axis: d-axis current increase mode
	CURRENT_STATE_ID_DOWN	4	Current status for the d axis: d-axis current decrease mode
	CURRENT_STATE_IQ_ZERO_CONST	0	Current status for the q axis: q-axis current always-0 mode
	CURRENT_STATE_IQ_SPEED_PI_OUTPUT	1	Current status for the q axis: q-axis command PI input mode
	CURRENT_VERR_COMP_LIMIT	(MOTOR_MCU_CFG_CARRIER_FREQ * INVERTER_CFG_DEADTIME/1000.0f)	Voltage error compensation period limiter value For details about MOTOR_MCU_CFG_CARRIER, see r_motor_module_cfg.h. For details about INVERTER_CFG_DEADTIME, see r_motor_inverter_cfg.h.

5.3.8 Adjustment and configuration of parameters

(a) Adjustment of the natural frequency and attenuation coefficient for current control

In the current control module, the control gain is adjusted by tuning the natural frequency for current control and the attenuation coefficient for current control. Set the natural frequency for current control in proportion to the frequency at which to perform current control. The natural frequency can be set to about 1/10 of the current control frequency. However, in many cases, a lower value may be set in consideration of noise during position detection and current detection.

For the attenuation coefficient for current control, a value in the range from 0.7 to 1.0 is ordinarily set. Setting a value nearer to 1.0 makes response more stable and moderate.

When you set or update the values of the natural frequency and attenuation coefficient for current control, use the following variables of the st_current_cfg_t structure (the structure for setting the parameters for controlling the current control module). After you have set the desired values in these variables, apply them by using R_MOTOR_CURRENT_ParameterUpdate (the API function for updating the parameters that control the current control module).

The natural frequency and attenuation coefficient for current control can be adjusted from RMW.

To set the natural frequency for current control, use f4_current_omega_hz. (See Table 5-20.)

To set the attenuation coefficient for current control, use f4_current_zeta. (See Table 5-20.)

(b) Setting the parameters for current control

Because the current control module uses the control interval and motor parameters, the control parameter configuration (R_MOTOR_CURRENT_ParameterUpdate) can be used to update the parameters. For details about the items that can be set, see the description of the st_current_cfg_t structure (structure for setting the parameters for controlling the current control module).

(c) Setting the initial values of the parameters for current control

The configurations of the current control module can be specified by using r_motor_module_cfg.h. The values set in this file are applied as initial values at system startup. For details about the items to be set, see 5.3.7 Macro definition.

5.4 Modulation (current control module)

In the sample program, the input voltage to the motor is generated by pulse width modulation (PWM). This module calculates the PWM duty ratio. A modulated voltage can be output to improve the efficiency of voltage usage. The modulation operation is set from the API of the current control module.

5.4.1 Description of the functionality

With this module, you can select from two types of pulse width modulation drive methods.

(a) Sine wave modulation (MOD_METHOD_SPWM)
The modulation factor m is defined as follows.

$$m = \frac{V}{E}$$

m : Modulation ratio V : Reference voltage E : Inverter input voltage

(b) Space Vector Modulation (MOD_METHOD_SVPWM) *
In vector control of a permanent magnet synchronous motor, generally, the desired voltage command value of each phase is generated sinusoidally. However, if the generated value is used as-is for the modulation wave for PWM generation, voltage utilization as applied to the motor (in terms of line voltage) is limited to a maximum of 86.7% with respect to inverter bus voltage. As such, as shown in the following expression, the average of the maximum and minimum values is calculated for the voltage command value of each phase, and the value obtained by subtracting the average from the voltage command value of each phase is used as the modulation wave. As a result, the maximum amplitude of the modulation wave is multiplied by $\sqrt{3}/2$, while voltage utilization becomes 100% and line voltage is unchanged.

$$\begin{pmatrix} V_u' \\ V_v' \\ V_w' \end{pmatrix} = \begin{pmatrix} V_u \\ V_v \\ V_w \end{pmatrix} + \Delta V \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\because \Delta V = -\frac{V_{max}+V_{min}}{2},\ V_{max} = max\{V_u, V_v, V_w\},\ V_{min} = min\{V_u, V_v, V_w\}$$

V_u, V_v, V_w: Command values of U-, V-, and W-phases

V_u', V_v', V_w': Command values of U-, V-, and W-phases for PWM generation (modulation wave)

The modulation factor m is defined as follows.

$$m = \frac{V'}{E}$$

m : Modulation ratio V' : Reference phase voltage for PWM
E : Inverter input voltage

5.4.2 Configurations

Table 5-22 lists the configurations for the modulation function.

Table 5-22 List of configurations

File name	Macro name	Settings	Description
r_motor_module_cfg.h	CURRENT_CFG_MODULATION_METHOD	MOD_METHOD_SVPWM	Pulse-width modulation drive mode

5.4.3 Structures

Table 5-23 lists the structures that are used for the modulation function.

Table 5-23 List of variables

Structure	Variable	Description
st_mod_t	f4_vdc	Bus voltage value [V]
	f4_1_div_vdc	1/f4_vdc
	f4_voltage_error_ratio	Voltage error ratio
	f4_max_duty	Maximum PWM duty cycle
	f4_min_duty	Minimum PWM duty cycle
	f4_neutral_duty	Intermediate value of the PWM duty cycle

5.4.4 Macro definition

Table 5-24 lists the macros that are used for the modulation function.

Table 5-24 List of macros

File name	Macro name	Defined value	Description
r_motor_current_modulation.h	MOD_DEFAULT_MAX_DUTY	1.0f	Maximum PWM duty cycle
	MOD_METHOD_SPWM	0	Pulse-width modulation drive mode: Sinusoidal PWM
	MOD_METHOD_SVPWM	1	Pulse-width modulation drive mode: Spatial vector PWM
	MOD_VDC_TO_VAMAX_MULT	0.6124f	Coefficient of the conversion from the input voltage to the maximum voltage
	MOD_SVPWM_MULT	1.155f	Coefficient for spatial vector PWM

5.4.5 Adjustment and configuration of parameters

There are no parameters to be set by the user for the modulation function.

5.5 Voltage error compensation (current control module)

The voltage error compensation function corrects for the effects of output voltage dead time. It operates through the API of the current control module.

5.5.1 Description of the functionality

In the voltage PWM converter, to prevent the switching elements of the upper and lower sides from creating a short circuit, a dead time during which the two elements are simultaneously turned off is set. Therefore, an error arises between the voltage command value and the voltage actually applied to the motor, degrading the control precision. Voltage error compensation is implemented to reduce this error.

The current dependency of the voltage error depends on the current (direction and magnitude), dead time, and the switching characteristics of the power elements to be used, and has the characteristics shown below. Voltage error compensation is achieved by applying the inverse voltage pattern of the voltage error (as shown below) to the voltage command value according to the current.

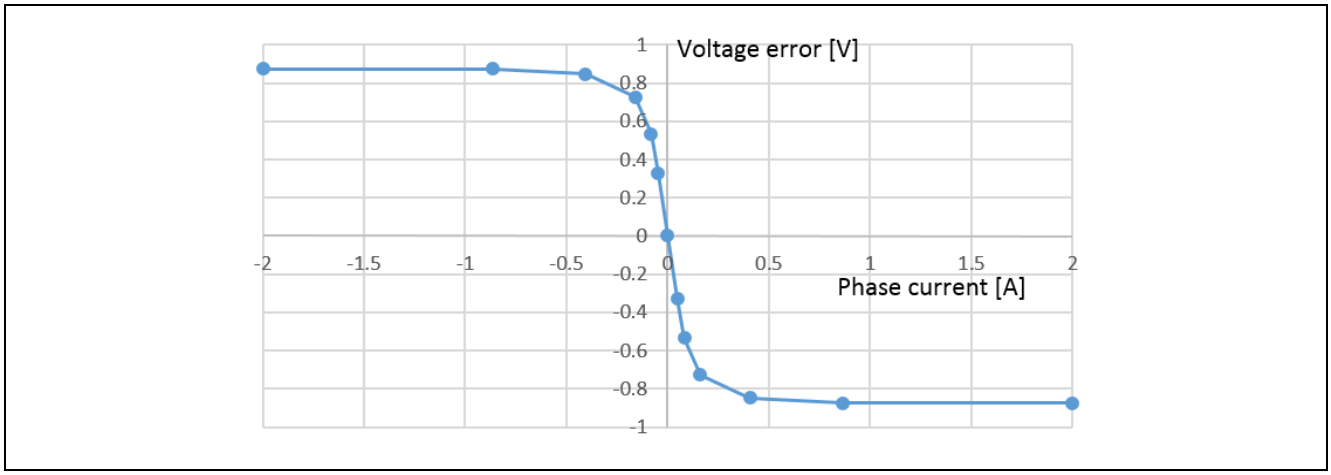


Figure 5-21 Current dependency in the voltage error (example)

5.5.2 Configurations

Table 5-25 lists the configurations for the voltage error compensation function.

Table 5-25 List of configurations

File name	Macro name	Settings	Description
r_motor_inverter_cfg.h	INVERTER_CFG_COMP_V0	RX26T: 0.564f RX14T: 0.234f	Voltage compensation table
	INVERTER_CFG_COMP_V1	RX26T: 0.782f RX14T: 0.391f	Voltage compensation table
	INVERTER_CFG_COMP_V2	RX26T: 0.937f RX14T: 0.463f	Voltage compensation table
	INVERTER_CFG_COMP_V3	RX26T: 1.027f RX14T: 0.508f	Voltage compensation table
	INVERTER_CFG_COMP_V4	RX26T: 1.058f RX14T: 0.524f	Voltage compensation table
	INVERTER_CFG_COMP_I0	RX26T: 0.022f RX14T: 0.035f	Voltage compensation table
	INVERTER_CFG_COMP_I1	RX26T: 0.038f RX14T: 0.066f	Voltage compensation table
	INVERTER_CFG_COMP_I2	RX26T: 0.088f RX14T: 0.131f	Voltage compensation table
	INVERTER_CFG_COMP_I3	RX26T: 0.248f RX14T: 0.313f	Voltage compensation table
	INVERTER_CFG_COMP_I4	RX26T: 0.865f RX14T: 0.609f	Voltage compensation table

5.5.3 Adjustment and configuration of parameters

(a) Setting the flag for whether to enable the voltage error compensation function

The voltage error compensation function is enabled by setting "u1_flag_volt_err_comp_use" (flag for whether to enable the voltage error compensation function) to MTR_FLG_SET when R_MOTOR_CURRENT_ParameterUpdate (setting of the control parameter for the current control module) is called. To disable the function, set this flag to MTR_FLG_CLR.

5.6 Speed control module

The speed control module controls the motor so that the speed follows the speed command. When receiving a speed command value, this module outputs a current command value accordingly. This module also controls the magnetic flux weakening of the submodules and the disturbance torque/speed estimation observer.

5.6.1 Functionality

Table 5-26 lists the functions of the speed control module.

Table 5-26 List of functions of the speed control module

Function	Description
Speed control	Calculates and outputs a current command value that allows the speed to follow the speed command value.
Speed command setting	Sets a speed command value in the speed control module.
Magnetic flux weakening control setting	Uses the magnetic flux weakening control to calculate and set the current command values for the d and q axes.
Disturbance torque/speed estimation observer setting	Estimates the speed by using the motor model, torque (q axis) current command value, and detected rotation speed.
IPD control switching	Switches the control mode from PID control to IPD control, which uses the IPD module.

5.6.2 Module configuration diagram

Figure 5-22 shows the module configuration of the speed control module.

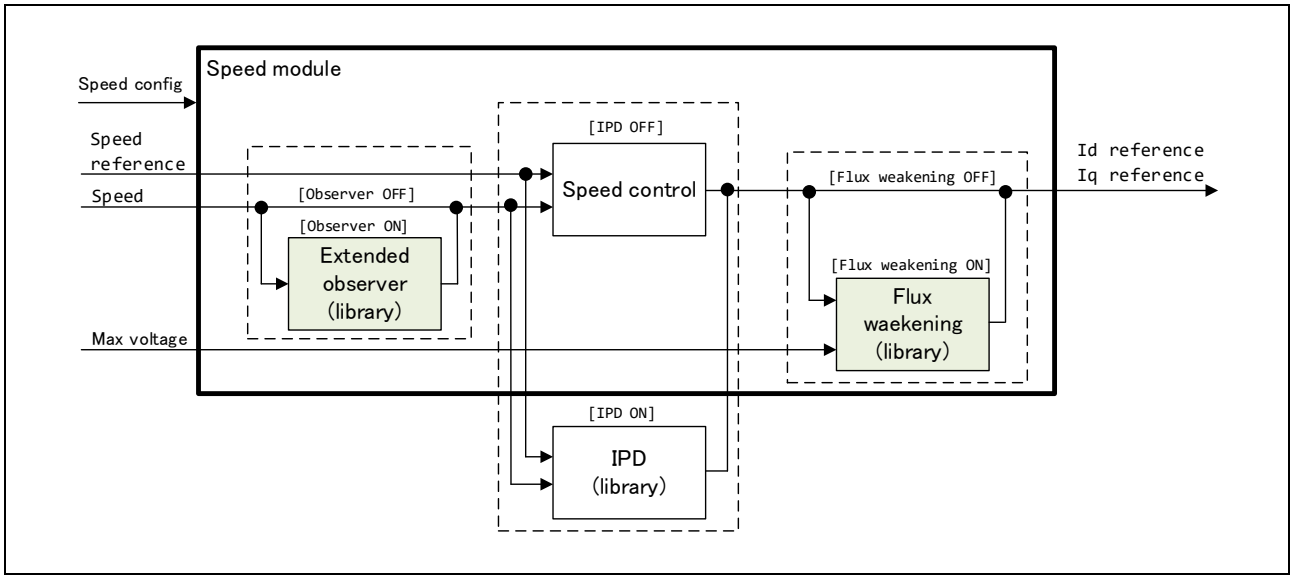


Figure 5-22 Speed control module configuration diagram

For details about the magnetic flux weakening control function and disturbance torque/speed estimation observer function, which are submodules of the speed control module, see 5.7 Magnetic flux weakening control (speed control module) and 5.8 Disturbance torque/speed estimation observer (speed control module).

5.6.3 Flowchart

Figure 5-23 shows the flowchart for speed control.

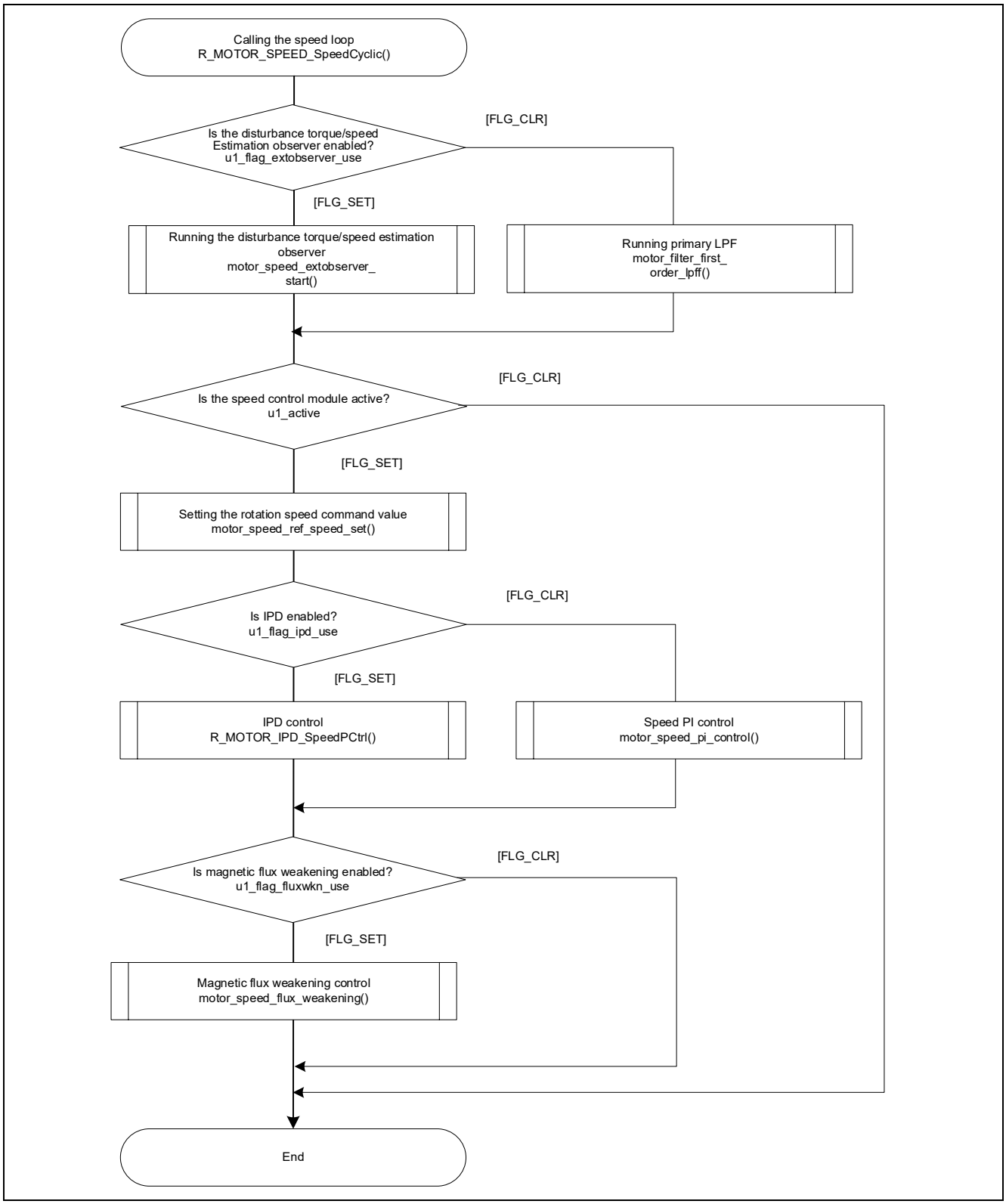


Figure 5-23 Flowchart for speed control

5.6.4 API function

Table 5-27 lists the API functions of the speed control module.

Table 5-27 List of API functions

API function	Description
R_MOTOR_SPEED_Open	Generates an instance of the speed control module.
R_MOTOR_SPEED_Close	Places the module in a reset state.
R_MOTOR_SPEED_Reset	Initializes the module.
R_MOTOR_SPEED_Run	Activates the module.
R_MOTOR_SPEED_ParameterSet	Inputs the variable information that is used for speed control.
R_MOTOR_SPEED_ParameterGet	Acquires the speed control results that are output.
R_MOTOR_SPEED_ParameterUpdate	Updates the control parameters of the module.
R_MOTOR_SPEED_SpdRefSet	Sets the speed command value.
R_MOTOR_SPEED_SpeedCyclic	Performs speed control.
R_MOTOR_SPEED_IPDInstanceAddressSet	Sets the address of the IPD module.
R_MOTOR_SPEED_IPDEnableSet	Enables the IPD module.
R_MOTOR_SPEED_ExtObserverParameterUpdate	Updates the control parameters for the disturbance torque/speed estimation observer.

5.6.5 Configurations

Table 5-28 lists the configurations for the speed control module. Set up the functions to be used and the necessary parameters. Table 5-29 shows the initial values.

Table 5-28 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	SPEED_CFG_FLUX_WEAKENING	Magnetic flux weakening control setting Enable: MTR_ENABLE Disable: MTR_DISABLE
	SPEED_CFG_OBSERVER	Disturbance torque/speed estimation observer setting Enable: MTR_ENABLE Disable: MTR_DISABLE
	SPEED_CFG_CTRL_PERIOD	Control interval setting [s]
	SPEED_CFG_OMEGA	Natural frequency for speed control [Hz]
	SPEED_CFG_ZETA	Attenuation coefficient for speed control
	SPEED_CFG_LPF_OMEGA	LPF bandwidth for speed control [Hz]
	SPEED_CFG_SPEED_LIMIT_RPM	Speed limit value [rpm]
	SPEED_CFG_RATE_LIMIT_RPM	Acceleration limit [rpm/s]
	SPEED_CFG_SOB_OMEGA	Natural frequency of the disturbance torque/speed estimation observer [Hz]

Table 5-29 List of initial values for configurations

Macro name	Settings
SPEED_CFG_FLUX_WEAKENING	MTR_DISABLE
SPEED_CFG_OBSERVER	MTR_DISABLE
SPEED_CFG_CTRL_PERIOD	RX26T: 0.0005f RX14T: 0.001f
SPEED_CFG_OMEGA	12.0f
SPEED_CFG_ZETA	1.0f
SPEED_CFG_LPF_OMEGA	250.0f
SPEED_CFG_SPEED_LIMIT_RPM	4500.0f
SPEED_CFG_RATE_LIMIT_RPM	1000.0f
SPEED_CFG_SOB_OMEGA	100.0f

5.6.6 Structure and variable information

Table 5-30 lists the structures and variables for the speed control module. For the speed control module, the structure for the speed control module (g_st_sc) is defined by securing an instance of the module from the API.

Table 5-30 List of structures and variables (1)

Structure	Variable	Description
st_speed_control_t Structure for the speed control module	u1_active	Selects whether to enable the module.
	u1_state_speed_ref	The variable for managing the states that determine the speed command value. It manages the states as shown in "Macro definition" below.
	u1_flag_extobserver_use	Flag for whether to use the disturbance torque/speed estimation observer
	u1_flag_ipd_use	Flag for whether to use the IPD module
	u1_flag_fluxwkn_use	Flag for whether to use magnetic flux weakening control
	f4_speed_ctrl_period	Speed loop control interval [s]
	f4_ref_speed_rad_ctrl	Speed command value for control [rad/s]
	f4_ref_speed_rad	Speed command value output by the position control module during position control [rad/s]
	f4_ref_speed_rad_manual	Speed command value set by the user during speed control [rad/s]
	f4_speed_rad_ctrl	Speed calculated by the speed control module [rad/s]
	f4_speed_rad	Speed that is input [rad/s]
	f4_max_speed_rad	Maximum speed [rad/s]
	f4_speed_rate_limit_rad	Speed variation limit value [rad/s]
	f4_speed_obsrv_rad	Speed calculated by the disturbance torque/speed estimation observer [rad/s]
	f4_id_ref_output	d-axis current command value [A]
	f4_iq_ref_output	q-axis current command value [A]
	f4_va_max	Maximum voltage on the d and q axes [V]
	f4_id_ad	d-axis current value [A]
	f4_iq_ad	q-axis current value [A]
	st_motor	Structure for motor constants
	st_pi_speed	Structure for PI control
	st_extobs	Structure for the disturbance torque/speed estimation observer
	st_fluxwkn	Structure for magnetic flux weakening control
	st_slpf	Structure for LPF
	*p_st_ipd	Structure for IPD

Table 5-31 List of structures and variables (2)

Structure	Variable	Description
st_speed_cfg_t Structure for setting the parameters for controlling the speed control module	u1_flag_extobserver_use	Flag for whether to use the disturbance torque/speed estimation observer
	u1_flag_fluxwkn_use	Flag for whether to use magnetic flux weakening control
	f4_max_speed_rpm	Maximum speed [rpm]
	f4_speed_ctrl_period	Speed control interval [s]
	f4_speed_rate_limit_rpm	Speed variation limit value [rpm]
	f4_speed_omega_hz	Natural frequency for speed control [Hz]
	f4_speed_zeta	Attenuation coefficient for speed control
	f4_speed_lpf_hz	LPF for speed control [Hz]
st_speed_input_t Structure for speed control module input	u1_state_speed_ref	Speed command status
	f4_ref_speed_rad	Speed command value [rad/s]
	f4_speed_rad	Speed that is to be input [rad/s]
	f4_va_max	Maximum voltage in the d and q axes [V]
st_speed_output_t Structure for speed control module output	f4_id_ref	d-axis current command value [A]
	f4_iq_ref	q-axis current command value [A]
	f4_ref_speed_rad_ctrl	Speed that is used for PI control [rad/s]
	f4_speed_rad_lpf	Speed after LPF [rad/s]
st_ext_observer_cfg_t Structure for setting the parameters for the disturbance observer	f4_extobs_omega	Natural frequency of the disturbance torque/speed estimation observer [Hz]

5.6.7 Macro definition

Table 5-32 lists the macros of the speed control module.

Table 5-32 List of macros

File name	Macro name	Defined value	Remarks
r_motor_speed_api.h	SPEED_STATE_ZERO_CONST	0	This macro is used to manage the state of the speed control module. The speed command value is always 0.
	SPEED_STATE_POSITION_CONTROL_OUTPUT	1	This macro is used to manage the state of the speed control module. The speed command value is used as the output of the position control module.
	SPEED_STATE_MANUAL	2	This macro is used to manage the state of the speed control module. The speed command value becomes the user-specified value.

5.6.8 Adjustment and configuration of parameters

(a) Adjustment of the natural frequency and attenuation coefficient for speed control

In the speed control module, the control gain is adjusted by tuning the natural frequency for speed control and the attenuation coefficient for speed control. Increasing the natural frequency for speed control improves the responsiveness, expanding the following capability of the speed to the commanded speed. The maximum settable natural frequency for speed control is limited to 1/3 of the maximum settable natural frequency for current control to prevent interference with current control. For the attenuation coefficient for speed control, a value in the range from 0.7 to 1.0 is ordinarily set. Setting a value nearer to 1.0 makes response more stable and moderate. Make adjustment while checking the speed responsiveness.

When you set or update the values of the natural frequency and attenuation coefficient for speed control, use the following variables of the st_speed_cfg_t structure (the structure for setting the parameters for controlling the speed control module). After you have set the desired values in these variables, apply them by using R_MOTOR_SPEED_ParameterUpdate (the API function for updating the parameters that control the speed control module).

- To set the natural frequency for speed control, use f4_speed_omega_hz. (See Table 5-30.)
- To set the attenuation coefficient for speed control, use f4_speed_zeta. (See Table 5-30.)

(b) Setting the parameters for speed control

Because the speed control module uses the control interval and motor parameters, the control parameter configuration (R_MOTOR_SPEED_ParameterUpdate) can be used to update the parameters. For details about the items that can be set, see the description of the st_speed_cfg_t structure (structure for setting the parameters for controlling the speed control module).

(c) Setting the initial values of the parameters for speed control

The configurations of the speed control module can be specified by using r_motor_module_cfg.h. The values set in this file are applied as initial values at system startup. For details about the items to be set, see 5.6.5.

5.7 Magnetic flux weakening control (speed control module)

The magnetic flux weakening control module is a submodule of the speed control module. When a motor that uses a magnet as the rotor rotates, an inductive voltage arises in proportion to the permanent magnet magnetic flux and rotation speed of the rotor. When the rotation speed increases and the inductive voltage becomes equal to the power supply voltage (that is, the voltage saturates), higher current can no longer flow into the motor, resulting in a saturated state that restricts any further increase in motor speed. Magnetic flux weakening control is a technology that solves this problem.

5.7.1 Description of the functionality

In magnetic flux weakening control, the d-axis current is applied in the negative direction to suppress the effect of voltage saturation due to induced voltage, thus enabling higher and more stable rotational speeds to be obtained.

In practice, the d-axis current is determined and controlled according to the formula shown in Figure 5-24.

$$I_d = \frac{-\psi_a + \sqrt{\left(\frac{V_{om}}{\omega}\right)^2 - \left(L_q I_q\right)^2}}{L_d}$$

$$\because V_{om} = V_{amax} - I_a R$$

V_{om}: Inductive voltage limit value [V]

V_{amax}: Maximum voltage vector value [V]

I_a: Current vector magnification [A]

Figure 5-24 Formula for calculating the d-axis command value in magnetic flux weakening control

5.7.2 Adjustment and configuration of parameters

There are no parameters to be set by the user for this module. To use this module, use R_MOTOR_SPEED_ParameterUpdate (API function for updating the control parameters for the speed control module) to set "u1_flag_fluxwkn_use" (flag for whether to use magnetic flux weakening control) to 1.

5.8 Disturbance torque/speed estimation observer (speed control module)

The disturbance torque/speed estimation observer module is a submodule of the speed control module. Enabling the observer function contributes to improved tracking of speed commands and reduced speed ripple.

5.8.1 Description of the functionality

This function provides software-based speed ripple reduction by implementing an observer-based speed estimation algorithm. The observer takes the torque and speed (ω) calculated from the q-axis command value (I_{q_ref}) as input, and obtains an estimated speed ($\hat{\omega}$) based on the plant model. The observer can reduce speed ripple and has less influence on the control system than ordinary filter processing. It is also possible to reduce the impact by the sensor's quantization error and speed ripple due to noise.

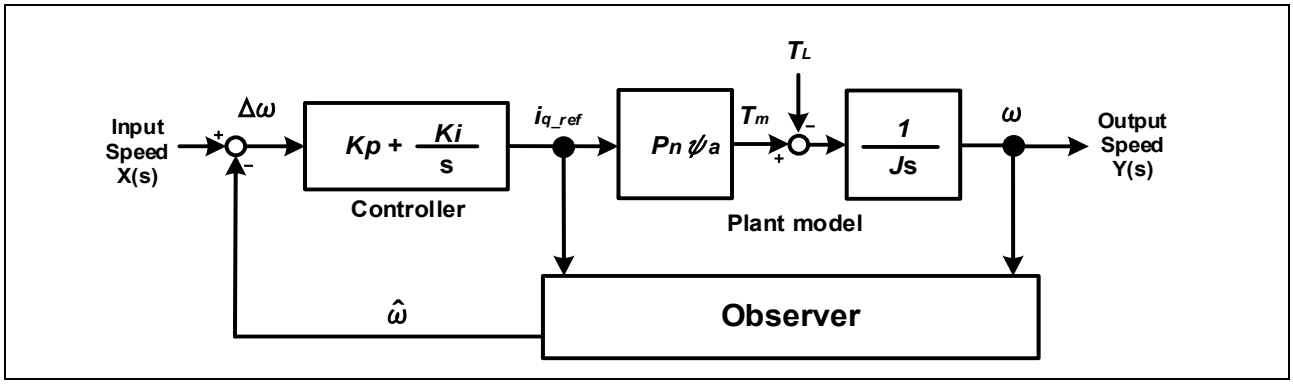


Figure 5-25 Speed control system model

5.8.2 Adjustment and configuration of parameters

This module sets the disturbance observer control parameters of the speed control module API by using R_MOTOR_SPEED_ExtObserverParameterUpdate (API function for updating the parameters). This module sets the following three types of parameters:

- Motor inertia
- Natural frequency of the disturbance observer
- Sampling interval of the observer

For the motor inertia and the sampling interval of the observer, make sure that you set correct values that are actually used for control. Decreasing the natural frequency for the disturbance observer further reduces speed ripple but degrades responsiveness to change of the commanded speed. Make adjustment while checking the speed responsiveness. As a guideline, the recommended natural frequency for the disturbance observer is about four to six times higher than the natural frequency for speed control.

5.9 Position control module

The position control module obtains the speed command value from the position command value and current position information. P control or IPD control can be selected as the position control mode.

In the sample program, position profile processing that determines the command value and drive mode (triangular or trapezoidal) for the position control from the deviation of the position command is implemented. Feedforward control for the speed command is also implemented to improve the responsiveness of speed control.

5.9.1 Functionality

Table 5-33 lists the functions of the position control module.

Table 5-33 List of functions of the position control module

Function	Description
Position control	Calculates and outputs a speed command value that allows the position to follow the position command value.
Speed feedforward control	Performs feedforward (FF) control to the speed command.
Position profile	Controls the position command value and drive mode (triangular or trapezoidal) based on the difference between the position command value and the actual position.
Dead band control	Detects whether the rotor position is in a dead band and adjusts the difference from the position command value.
IPD control switching	Switches the position control mode to IPD control.

5.9.2 Module configuration

Figure 5-26 shows the module configuration of the position control module.

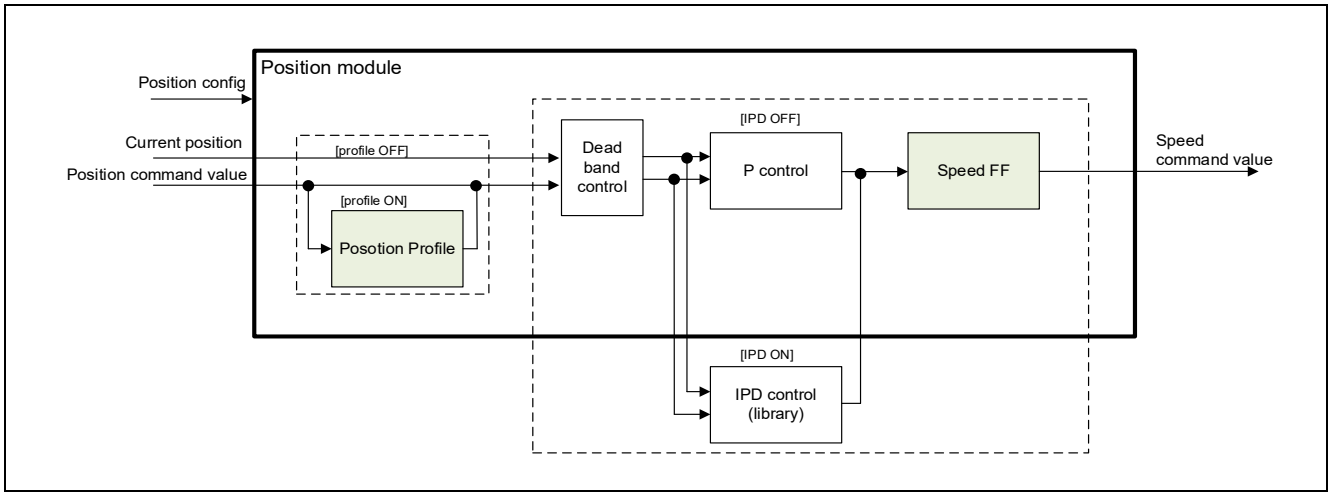


Figure 5-26 Position control module configuration diagram

5.9.3 Flowcharts

Figure 5-27 shows the flowchart for position control.

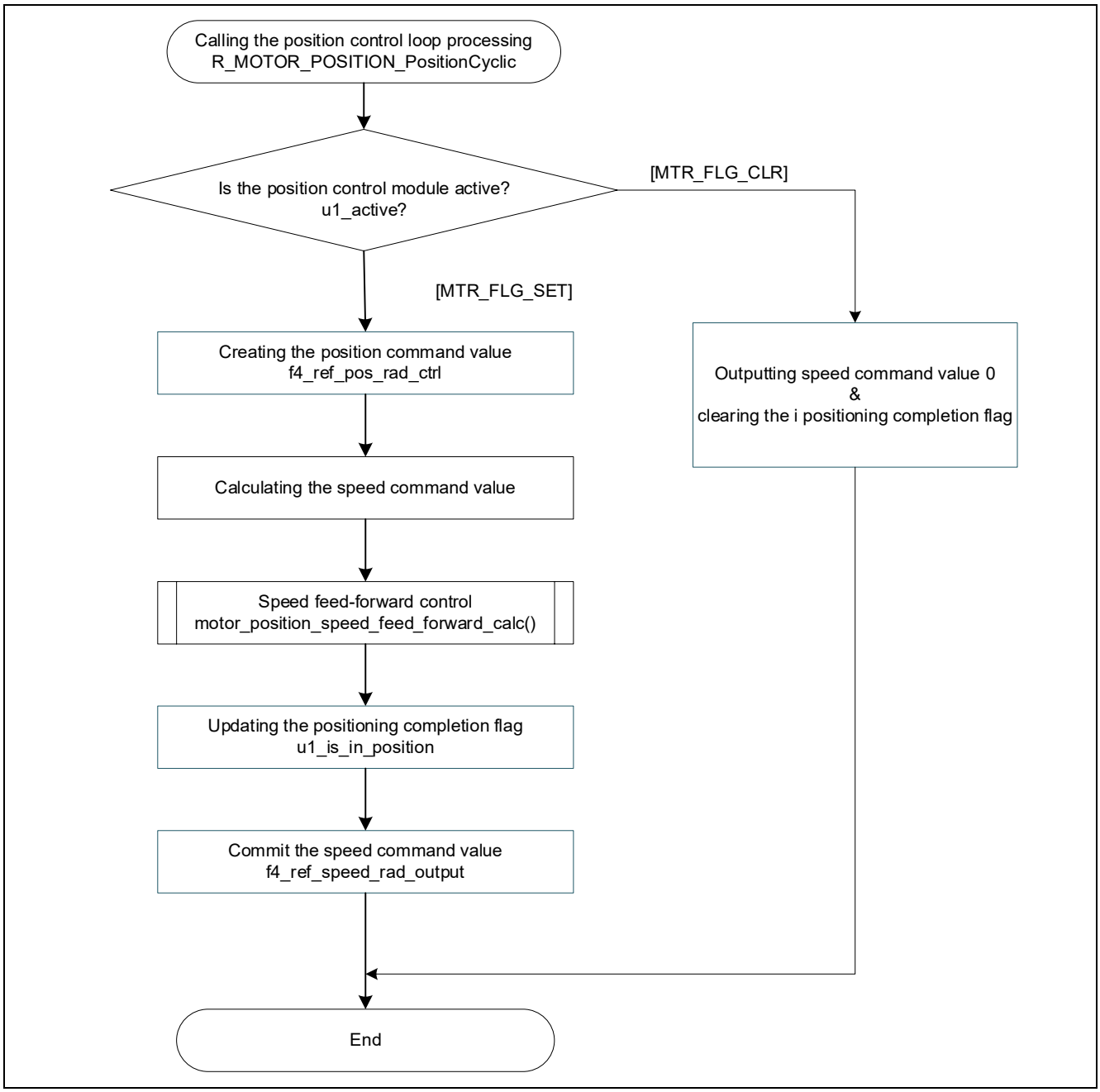


Figure 5-27 Flowchart for the position control loop processing

For a detailed flow for the position command creation processing, see Figure 5-28.

For a detailed flow for the speed command value calculation processing, see Figure 5-29.

For the modes shown in the detailed flows, see 5.9.4.

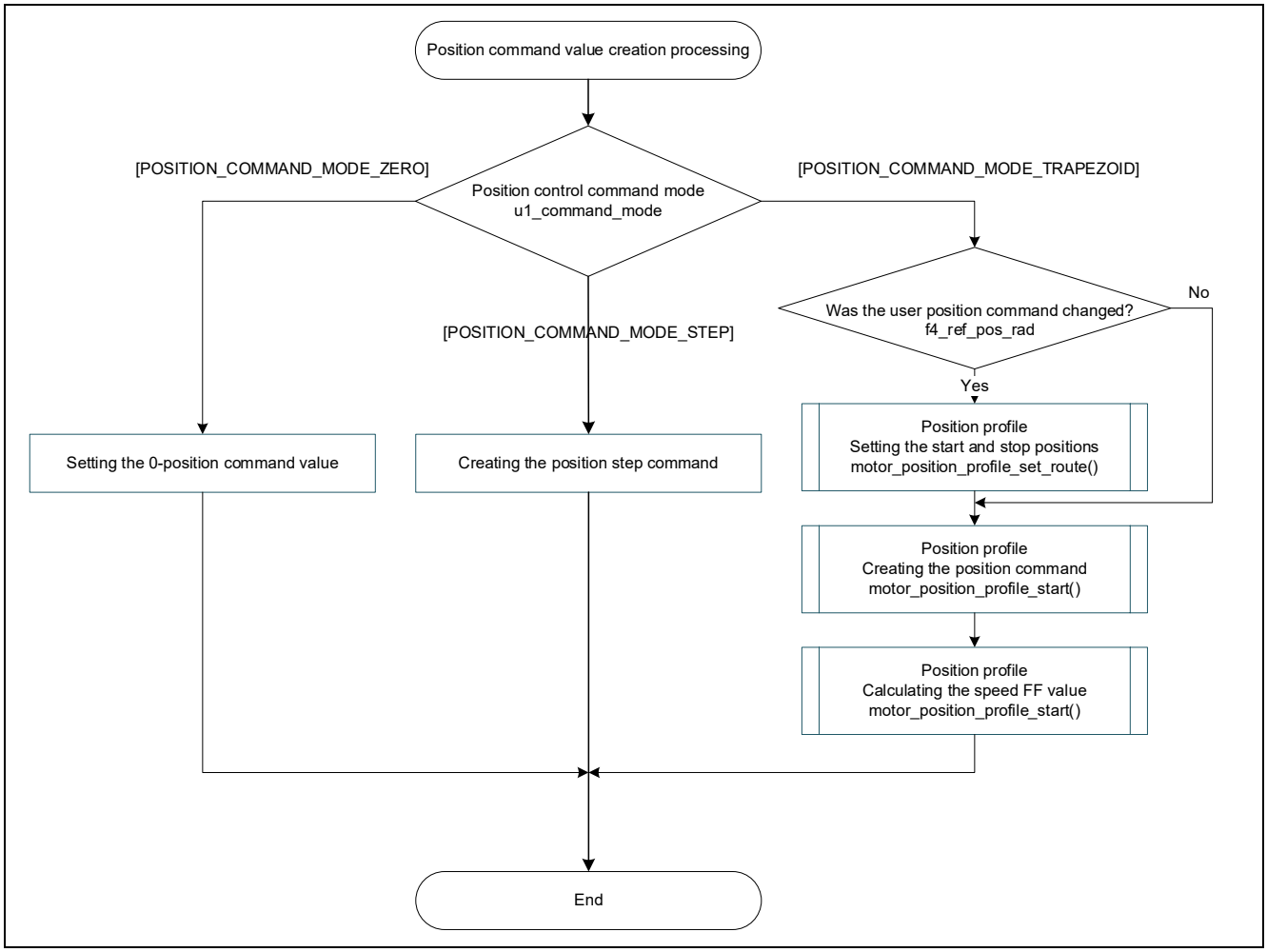


Figure 5-28 Flowchart for the position command creation processing

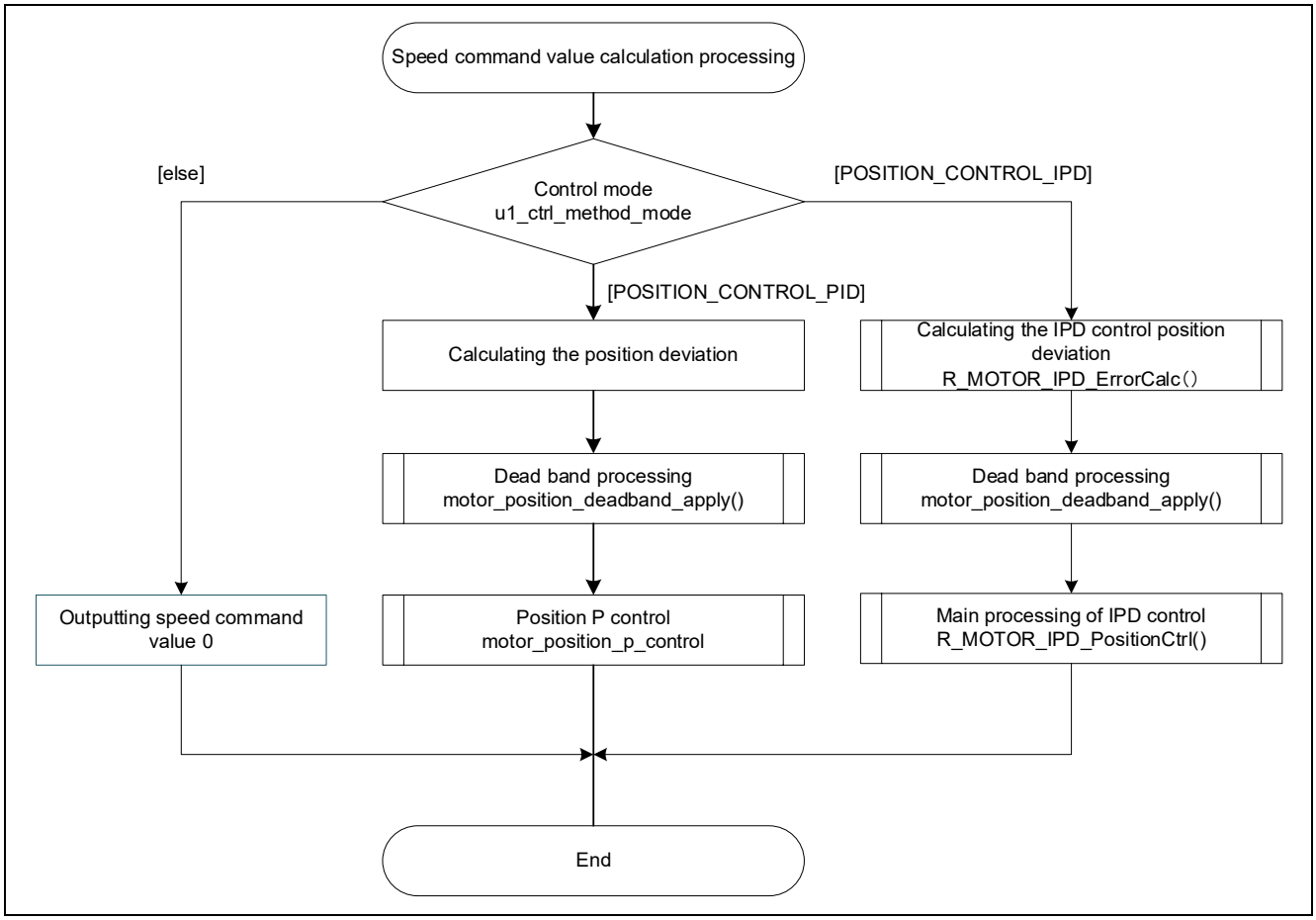


Figure 5-29 Speed command value calculation processing

5.9.4 Mode management

(a) Position command modes

Table 5-34 lists the position command modes.

Table 5-34 List of position control command modes

Mode name	Description
POSITION_COMMAND_MODE_ZERO	Position always-0 mode
POSITION_COMMAND_MODE_STEP	Step mode
POSITION_COMMAND_MODE_TRAPEZOID	Speed trapezoidal wave mode

The mode is switched by using R_MOTOR_POSITION_CommandModeSet (API function for setting the position control command mode).

(b) Control modes

Table 5-35 lists the control modes.

Table 5-35 List of control modes

Mode name	Remarks
POSITION_CONTROL_PID	PID control
POSITION_CONTROL_IPD	IPD control

The mode is switched by using R_MOTOR_POSITION_IPDEnableSet (API function for enabling the IPD module).

5.9.5 API function

Table 5-36 lists the API functions for the position control module.

Table 5-36 List of API functions

API function	Description
R_MOTOR_POSITION_Open	Generates an instance of the position control module.
R_MOTOR_POSITION_Close	Places the position control module in a reset state.
R_MOTOR_POSITION_Reset	Initializes the position control module.
R_MOTOR_POSITION_Run	Activates the position control module.
R_MOTOR_POSITION_PositionCyclic	Performs the position control loop processing.
R_MOTOR_POSITION_ParameterSet	Sets the parameters that are used for the position control loop.
R_MOTOR_POSITION_ParameterGet	Acquires the variable information of the position control module.
R_MOTOR_POSITION_ParameterUpdate	Updates the control parameters of the position control module.
R_MOTOR_POSITION_PosRefSet	Sets the position command.
R_MOTOR_POSITION_CommandModeSet	Sets the position control command mode.
R_MOTOR_POSITION_IPDInstanceAddressSet	Registers the address of the instance generated by the IPD module.
R_MOTOR_POSITION_IPDEnableSet	Enables a call to the IPD module.
R_MOTOR_POSITION_Sync	Changes the position information. For example, this item is used to resume the processing from the position at which the processing was stopped previously.

5.9.6 Configurations

Table 5-37 shows the configurations that are used for the position control module. Set up the functions to be used and the necessary parameters. Table 5-38 shows the initial values.

Table 5-37 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	POSITION_CFG_INTERVAL_TIME	Position control interval [s]
	POSITION_CFG_SPEED_FF_RATIO	Speed feed-forward proportionality coefficient
	POSITION_CFG_DEAD_BAND	Dead band (number of position sensor pulses)
	POSITION_CFG_INTERVAL_TIME	Stationary wait time for position response (number of control intervals)
	POSITION_CFG_OMEGA	Natural frequency for position control [Hz]
	POSITION_CFG_BAND_LIMIT	Range in which the position error is zero (number of position sensor pulses)

Table 5-38 List of initial values for configurations

Macro name	Settings
POSITION_CFG_CTRL_PERIOD	SPEED_CFG_CTRL_PERIOD
POSITION_CFG_SPEED_FF_RATIO	0.8f
POSITION_CFG_DEAD_BAND	1.0f
POSITION_CFG_INTERVAL_TIME	RX26T: 800.0f RX14T: 400.0f
POSITION_CFG_OMEGA	4.0f
POSITION_CFG_BAND_LIMIT	3.0f

5.9.7 Structure and variable information

Table 5-39 lists the structures that are used for the position control module. For the position control module, the structure for the position control module (g_st_pc) is defined by securing an instance of the module from the API.

Table 5-39 List of variables

Structure	Variable	Description
st_motor_position_t Structure for the position control module	u1_is_in_position	Positioning completion flag
	u1_active	Flag indicating whether the module is active
	u1_pos_command_mode	Position command value creation mode
	u1_ctrl_method_mode	IPD/P control mode switching
	u2_pos_dead_band	Dead band (number of position sensor pulses)
	u2_pos_band_limit	Positioning completion width (number of position sensor pulses)
	f4_pos_kp	Position P control gain coefficient
	f4_pos_err_rad	Position deviation [rad]
	f4_pos_rad	Current position [rad]
	f4_ref_pos_rad	Position command value (command from the upper layer) [rad]
	f4_ref_pos_pre_rad	Previous position command value [rad]
	f4_ref_pos_rad_ctrl	Position command after the position profile is processed [rad]
	f4_speed_ff_rad	Speed FF value [rad/s]
	f4_speed_ff_ratio	Speed feed-forward proportionality coefficient
	f4_ref_speed_rad_output	Speed command value [rad/s]
	f4_max_speed_rad	Maximum speed [rad/s]
	f4_ctrl_period	Control interval [s]
	f4_mech_angle_per_sensor_cnt	Angle per position sensor count [rad]
	st_ppf	Structure for the position profile
	st_motor	Structure for motor constants
	*p_st_ipd	IPD module generation instance pointer
st_position_cfg_t Structure for setting the parameters for controlling the position control module	st_motor	Structure for motor constants
	u2_dead_band	Dead band (number of position sensor pulses)
	u2_band_limit	Positioning completion width (number of position sensor pulses)
	u2_pos_interval_time	Time to wait for stability (number of control intervals)
	f4_feedforward_ratio	Speed feed-forward proportionality coefficient

Structure	Variable	Description
st_position_cfg_t Structure for setting the parameters for controlling the position control module	f4_position_omega_hz	Frequency for position control [Hz]
	f4_ctrl_period	Control interval [s]
	f4_mech_angle_per_sensor_cnt	Angle per position sensor pulse [rad]
	f4_max_speed_rad	Maximum speed [rad/s]
	f4_accel_time	Acceleration time [s]
	f4_posprof_max_speed_rad	Mechanical maximum speed for position profiling [rad/s]
st_position_input_t Structure for position control module input	f4_position_rad	Current position [rad]
st_position_output_t Structure for position control module output	f4_speed_ref	Speed command output value [rad/s]
	f4_position_err	Position deviation value [rad] Use this variable when you want to perform deviation decision externally in cases such as when automatic adjustment is used.
	u1_in_position	Positioning completion flag

5.9.8 Macro definition

Table 5-40 lists the macros that are used for the position control module.

Table 5-40 List of macros

File name	Macro name	Defined value	Remarks
r_motor_position_api.h	POSITION_COMMAND_MODE_ZERO	0	Position command mode: Position 0 mode
	POSITION_COMMAND_MODE_STEP	1	Position command mode: Step mode
	POSITION_COMMAND_MODE_TRAPEZOID	2	Position command mode: Speed trapezoidal wave mode
	POSITION_CONTROL_PID	0	Control mode: PID control
	POSITION_CONTROL_IPD	1	Control mode: IPD control

5.9.9 Parameter adjustment and configuration methods

(a) Adjusting the natural frequency for position control

In the position control module, the natural frequency for position control is tuned to adjust the gain of P control. The maximum settable natural frequency for position control is 1/3 of the maximum settable natural frequency for speed control.

When you set or update the values of the natural frequencies for control use, use the following variables of the st_position_cfg_t structure (the structure for setting the parameters for controlling the position control module). After you have set the desired values in these variables, apply them by using R_MOTOR_POSITION_ParameterUpdate (the API function for updating the parameters that control the position controlling the position control module).

- To set the natural frequency for position control, use f4_posprof_max_speed_rad. (See Table 5-30.)

(b) Setting the parameters for position control

Because the position control module uses the control interval and motor parameters, the control parameter configuration (R_MOTOR_POSITION_ParameterUpdate) can be used to update the parameters. For details about the items that can be set, see the description of the st_position_cfg_t structure (structure for setting the parameters for controlling the position control module).

(c) Setting the initial values of the parameters for position control

The configurations of the position control module can be specified by using r_motor_module_cfg.h. The values set in this file are applied as initial values at system startup. For details about the items to be set, see 5.9.6 Configurations.

5.10 Position profile (position control module)

The position profile function operates through the API of the position control module.

5.10.1 Description of the functionality

The position profile has a function that controls the speed command value by recalculating the speed command value at each control interval based on the position command value that is preset, the acceleration/deceleration time, and the maximum speed (moving-average acceleration/deceleration algorithm). Figure 5-30 shows an overview of this algorithm. If the speed obtained from the position deviation and acceleration time exceeds the maximum speed, a position command value is created so that the speed command value forms a trapezoidal shape. For the variables and other items shown in Figure 5-30, see Table 5-11 and Table 5-12.

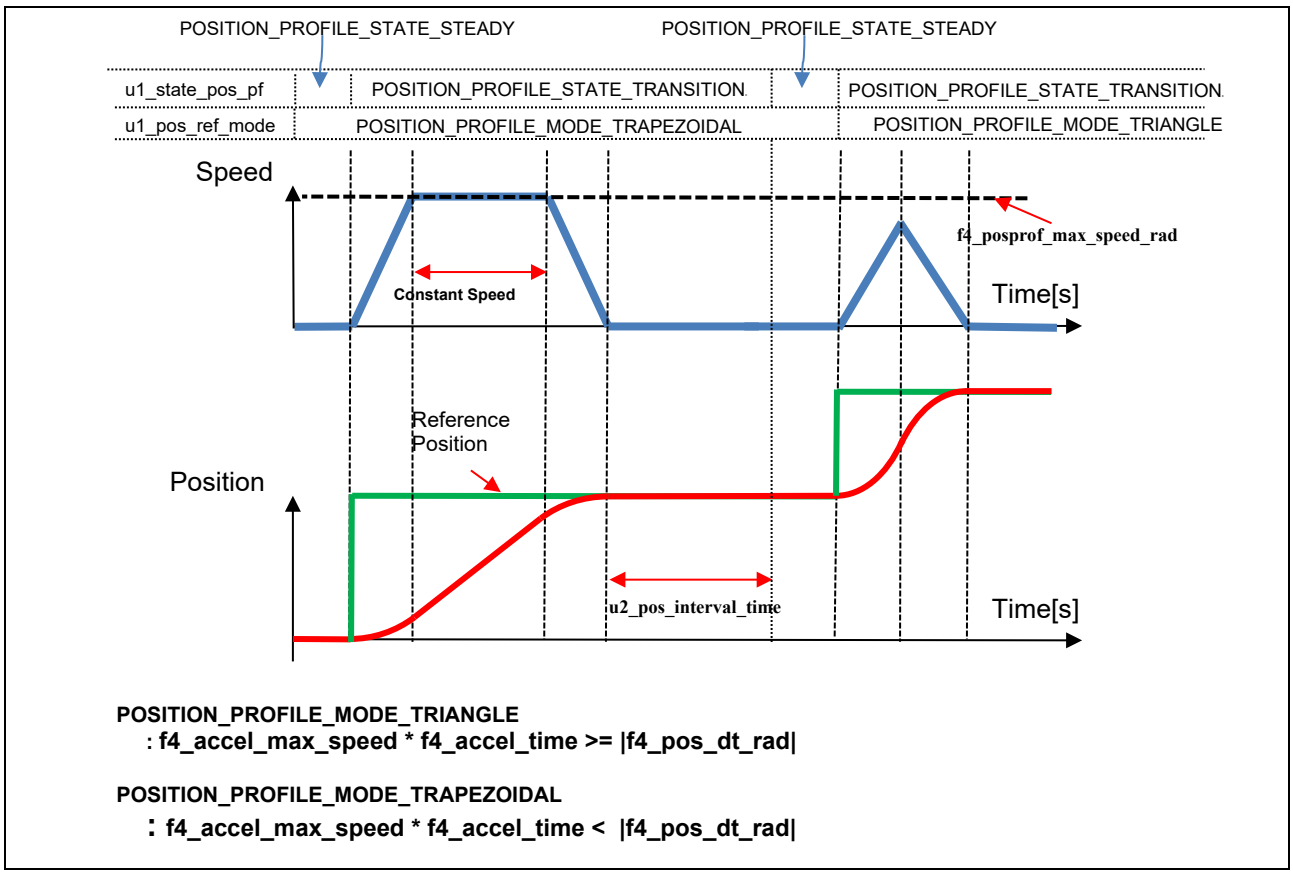


Figure 5-30 Shaping of the position command values and triangular/trapezoidal speed command values

5.10.2 Configurations

Table 5-41 shows the configurations that are used for the position profile function. Set up the functions to be used and the necessary parameters. Table 5-42 shows the initial values.

Table 5-41 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	POSITION_CFG_CTRL_PERIOD	Position control interval [s]
	POSITION_CFG_INTERVAL_TIME	Stationary wait time for position response (number of position sensor pulses)
r_motor_targetmotor_cfg.h	MOTOR_CFG_MAX_SPEED_RPM	Maximum speed [rpm]

Table 5-42 List of initial values for configurations

Macro name	Settings
POSITION_CFG_CTRL_PERIOD	SPEED_CFG_CTRL_PERIOD
POSITION_CFG_INTERVAL_TIME	RX26T: 800.0f RX14T: 400.0f
MOTOR_CFG_MAX_SPEED_RPM	4000.0f

5.10.3 Structures

Table 5-43 lists the structures that are used for the position profile function. The values set in a structure can be checked by using the variable for position control module management. See also Table 5-30. "st_ppf" in this table is the relevant variable.

Table 5-43 List of variables

Structure	Variable	Description
st_position_profiling_t Structure for the position profile	u1_state_pos_pf	Position profile status
	u1_pos_ref_mode	Position command mode
	u2_interval_time	Profile interval (number of control intervals)
	u2_interval_time_buff	Profile interval buffer (number of control intervals)
	u2_interval_time_cnt	Profile interval counter
	f4_ctrl_period	Control period [s]
	f4_accel_time	Acceleration time [s]
	f4_accel_time_buff	Acceleration time buffer [s]
	f4_accel_time_inv	Inverse number of the acceleration time
	f4_max_accel_time	Maximum acceleration time [rad/s]
	f4_accel_max_speed	Maximum acceleration speed [rad/s]
	f4_accel_max_speed_buff	Maximum acceleration speed buffer [rad/s]
	f4_time_sec	Timer counter for profiling use
	f4_pos_st_rad	Start position [rad]
	f4_pos_ed_rad	End position [rad]
	f4_pos_dt_rad	Profile position error [rad]
	f4_pos_dt_time_sec	Position error/Maximum speed [s]

5.10.4 Macro definition

Table 5-44 lists the macros that are used for the position profile function.

Table 5-44 List of macros

File name	Macro name	Defined value	Remarks
r_motor_position_profiling.h	POSITION_PROFILE_ACCEL_TIME	0.3f	Acceleration time for the speed command value [s]
	POSITION_PROFILE_CTRL_TRIANGLE	0	Triangular wave control mode
	POSITION_PROFILE_CTRL_TRAPEZOIDAL	1	Trapezoidal wave control mode
	POSITION_PROFILE_STEADY_STATE	0	Steady state
	POSITION_PROFILE_TRANSITION_STATE	1	Transition state

5.10.5 Adjustment and configuration of parameters

(a) Setting the parameters for control use

A position command value with acceleration/deceleration time consideration can be created by using R_MOTOR_POSITION_ParameterUpdate (the API function for updating the parameters that control the position control module) to set the following variables. Note that these are variables that adjust the operation shown in Figure 5-30.

- Acceleration time: f4_accel_time
- Maximum speed: f4_posprof_max_speed_rad
- Time to wait for stabilization: u2_pos_interval_time

5.11 IPD module

5.11.1 Functionality

In position control, if the resolutions of the position and speed are low, continuous vibration occurs during positioning. This problem occurs because the system cannot respond to small variations in position deviation. Reduction of vibration that occurs during positioning requires an integrating element that works to accumulate small variations and clear the deviation to zero.

The IPD controller adopts a control method in which only integration acts on deviation, and proportion and derivation act on only the operation amount (controller output). With this method, vibration that occurs during positioning can be reduced even with increased responsiveness.

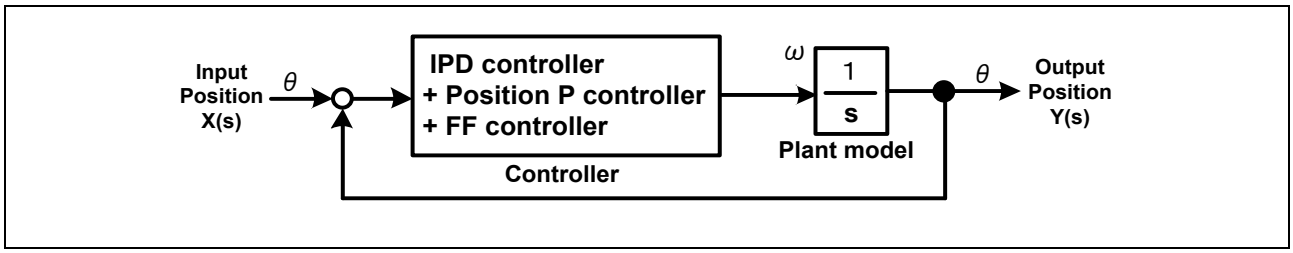


Figure 5-31 Model of IPD control (position control)

The IPD control that is actually implemented is combined with normal proportion control and feedforward control.

5.11.2 API function

Table 5-45 lists the API functions for the IPD module.

Table 5-45 List of API functions

API function	Description
R_MOTOR_IPD_Open	Generates an instance of the IPD module.
R_MOTOR_IPD_Close	Places the module in a reset state.
R_MOTOR_IPD_Reset	Initializes the module.
R_MOTOR_IPD_ParameterUpdate	Updates the control parameters of the module.
R_MOTOR_IPD_CtrlGainCalc	Calculates the gain.
R_MOTOR_IPD_SpeedPCtrl	Performs speed control for the IPD module.
R_MOTOR_IPD_ErrorCalc	Calculates the deviation of the position that will be used for control.
R_MOTOR_IPD_PositionCtrl	Performs position control for the IPD module.
R_MOTOR_IPD_PositionSync	Updates the IPD internal position variable information.

5.11.3 Structure and variable information

Table 5-46 lists the structures and variables for the IPD module. In the IPD module, the structure for the IPD module (g_st_ipd) is defined by securing an instance of the module from the API.

Table 5-46 List of structures and variables

Structure	Variable	Description
st_ipd_ctrl_t Structure for the IPD module	u1_ipd_lpf_flag	Flag for whether LPF is enabled
	f4_ref_pos_pre_rad_ctrl	Previous position command value [rad]
	f4_ipd_pos_k	Position control gain coefficient
	f4_ipd_pos_1st_fb_rad	Position feed-back (FB, hereinafter) value [rad]
	f4_ipd_pos_1st_fb_pre_rad	Previous position FB value [rad]
	f4_ipd_pos_2nd_fb_rad	Position FB value [rad] (stage 2)
	f4_ipd_ref_pos_rad	Position command value [rad]
	f4_ipd_err_rad	Position FB deviation value [rad]
	f4_ipd_pos_fb_k	Position FB gain coefficient
	f4_ipd_pos_ff_rad	Position feed-forward value [rad]
	f4_ipd_pos_ff_k	Position feed-forward gain coefficient
	f4_ipd_pos_p_rad	Position P control value [rad]
	f4_ipd_pos_kp	Position P control gain coefficient
	f4_ipd_pos_kp_ratio	Position P control amount ratio
	f4_ipd_pos_ff_ratio	Position feed-forward gain ratio
	f4_ipd_speed_k	Speed gain coefficient
	f4_ipd_speed_k_ratio	Speed gain ratio
	f4_ipd_ref_speed_rad	Speed command value [rad]
	f4_ipd_err_input_limit	Position deviation limiter [rad]
	f4_ipd_err_integrator_limit	Position error integrator limiter coefficient
	f4_ipd_lpf_omega	LPF natural frequency [Hz]
	f4_ipd_lpf_zeta	LPF attenuation coefficient
	st_ipd_2nd_lpf	Secondary LPF structure

5.11.4 Macro definition

Table 5-47 lists the macros for the IPD module.

Table 5-47 List of macros

File name	Macro name	Defined value	Remarks
r_motor_ipd_api.h	IPD_LPF_OMEGA	200.0f	Natural frequency for position LPF [Hz]
	IPD_LPF_ZETA	1.0f	Attenuation coefficient for position LPF
	IPD_SPEED_RATIO	2.5f	Speed gain coefficient
	IPD_POS_FF_RATIO	0.9f	Position FF coefficient
	IPD_POS_KP_RATIO	0.3f	Position gain coefficient
	IPD_POS_ERR_INPUT_LIMIT	10.0f	Position deviation limiter [rad]
	IPD_POS_ERR_INTEGRATOR_LIMIT_RATIO	1.0f	Position deviation integrator limiter coefficient
	IPD_LPF_FLAG	IPD_LPF_ON	Flag for whether LPF is enabled

5.11.5 Adjustment and configuration of parameters

(a) Setting operation coefficients

Use R_MOTOR_IPD_ParameterUpdate (the API function for updating control parameters) to set operation coefficients. Table 5-48 shows the values that are set in the sample program.

Table 5-48 Parameter setting example

API argument	Description	Value to be set when the manager module is called
f4_ipd_pos_kp_ratio	Position gain coefficient	IPD_POS_KP_RATIO
f4_ipd_pos_ff_ratio	Position FF coefficient	IPD_POS_FF_RATIO
f4_ipd_speed_k_ratio	Speed gain coefficient	IPD_SPEED_RATIO
f4_ipd_err_input_limit	Position deviation limiter	IPD_POS_ERR_INPUT_LIMIT
f4_ipd_err_integrator_limit	Position deviation integrator limiter coefficient	IPD_POS_ERR_INTEGRATOR_LIMIT_RATIO

5.12 Sensor module (encoder)

The sensor module calculates the position and speed of the motor. In the sample program, the sensor module for the encoder calculates the position and speed from encoder signals and outputs the calculation results. This module also supports startup using the Hall sensor input. This startup can be enabled by specifying the relevant configuration.

5.12.1 Functionality

Table 5-49 lists the functions of the sensor module.

Table 5-49 List of functions of the sensor module

Function	Description
Position information acquisition	Acquires the rotor position information of the motor.
Speed information acquisition	Acquires the rotation speed of the motor.

5.12.2 Module configuration diagram

Figure 5-32 shows the module configuration.

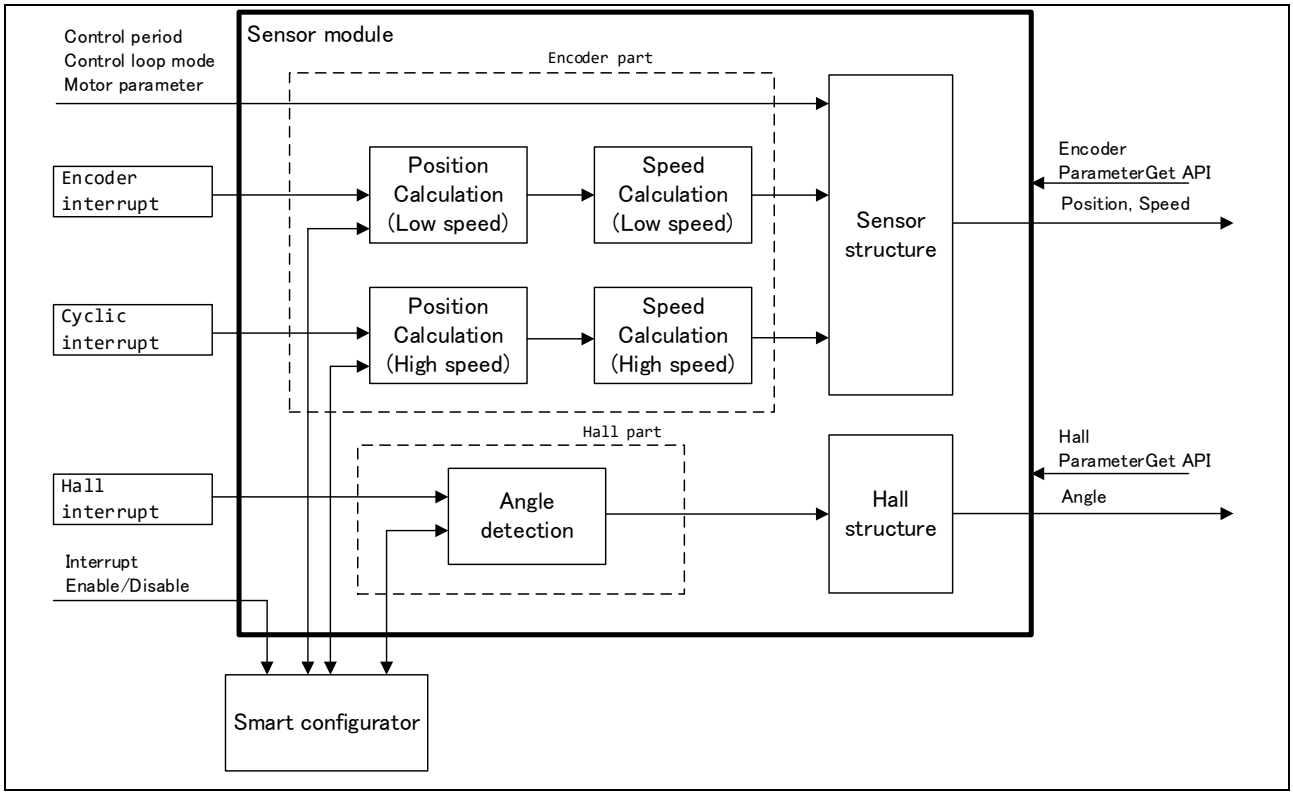


Figure 5-32 Sensor module configuration diagram

5.12.3 Flowchart

Figure 5-33 shows the flowchart for calculating the position and speed from input capture interrupts of the encoder.

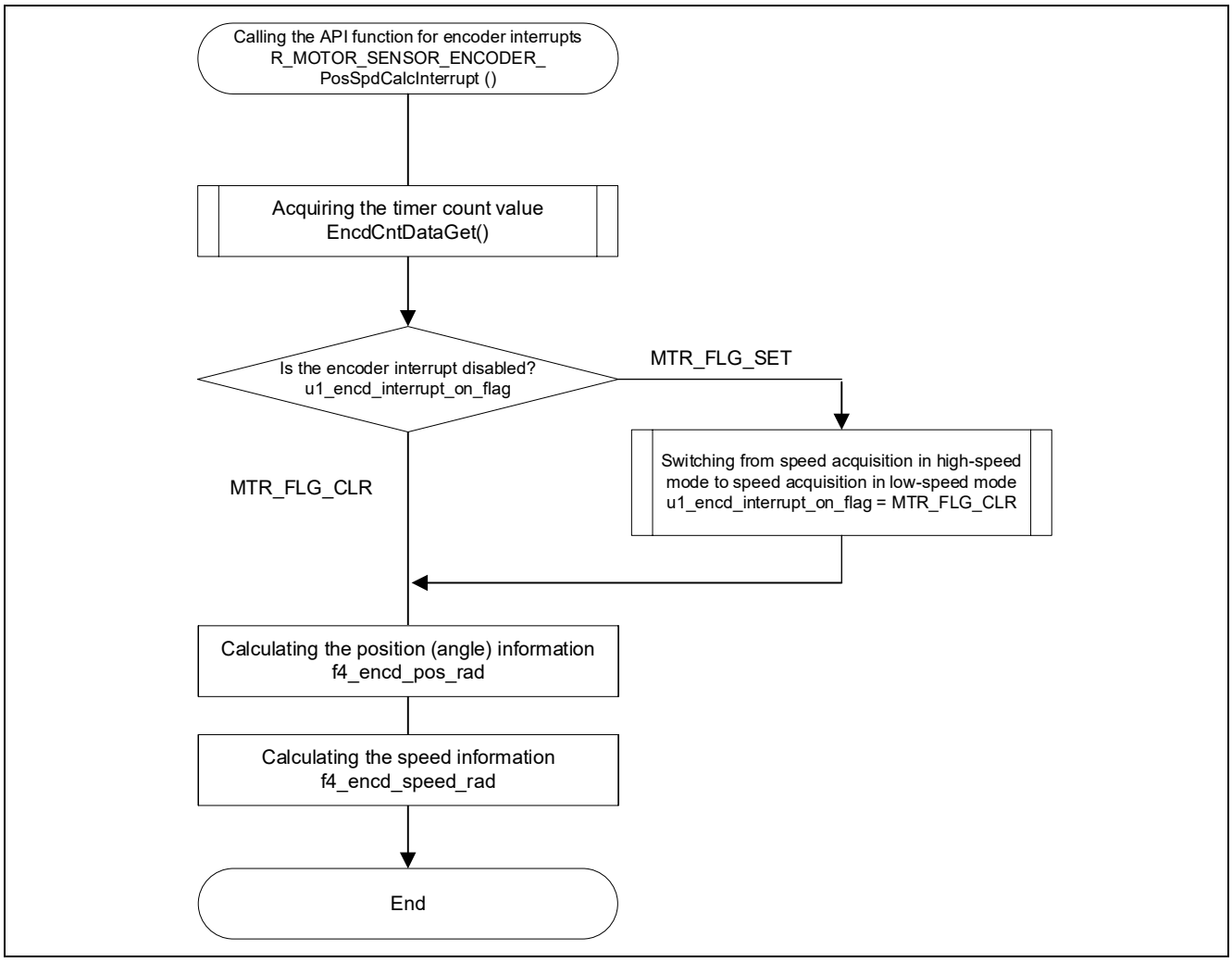


Figure 5-33 Encoder interrupt processing flowchart

Figure 5-34 shows the flowchart for the interrupt processing that is performed at startup using Hall sensor interrupts.

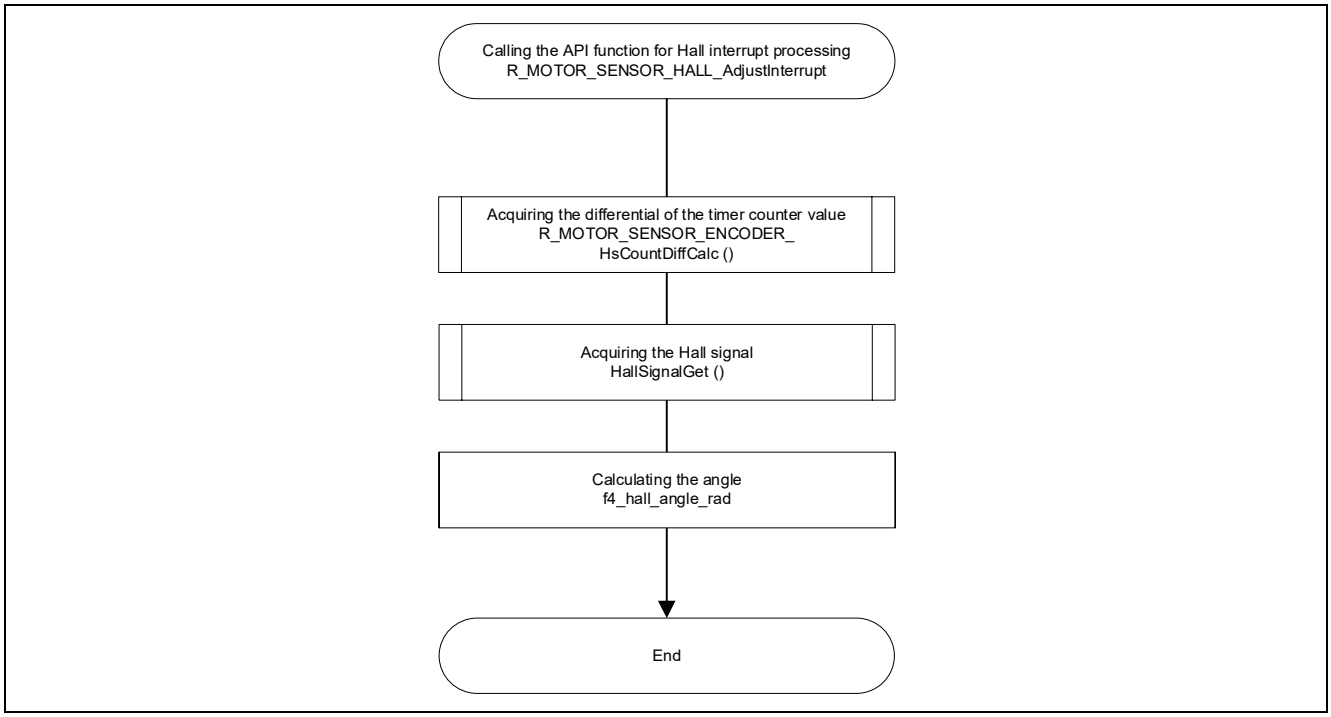


Figure 5-34 Hall sensor interrupt processing flowchart

5.12.4 API

Table 5-50 lists the API functions for the sensor module.

Table 5-50 List of API functions

API function	Description
R_MOTOR_SENSOR_ENCODER_Open	Generates an instance of the sensor module. Run this API function first when using this module.
R_MOTOR_SENSOR_ENCODER_Close	Places the sensor module in a reset state.
R_MOTOR_SENSOR_ENCODER_Reset	Initializes the module.
R_MOTOR_SENSOR_ENCODER_ParameterGet	Acquires the position and speed information of the encoder.
R_MOTOR_SENSOR_ENCODER_ParameterUpdate	Updates the control parameters of the sensor module.
R_MOTOR_SENSOR_ENCODER_DriverParameterUpdate	Sets the corresponding Smart Configurator functions in the sensor module.
R_MOTOR_SENSOR_ENCODER_PosSpdCalcInterrupt	Calculates the position and speed from the interval of signal interrupts of the encoder.
R_MOTOR_SENSOR_ENCODER_PosSpdCalc	Calculates the position and speed from the encoder pulse information that was input within a certain number of intervals.
R_MOTOR_SENSOR_ENCODER_HsCountDiffCalc	Acquires the differential of the value of the timer that counts the number of encoder pulses.
R_MOTOR_SENSOR_HALL_ParameterGet	Acquires the angle information of the Hall sensor.
R_MOTOR_SENSOR_HALL_AngleAdjustInit	Acquires the initial signal of the Hall sensor.
R_MOTOR_SENSOR_HALL_AdjustInterrupt	Checks the energization pattern of the Hall sensor.
R_MOTOR_SENSOR_HALL_InterruptEnable	Enables Hall signal interrupts.
R_MOTOR_SENSOR_HALL_InterruptDisable	Disables Hall signal interrupts.

5.12.5 Configurations

Table 5-51 lists the configurations for the sensor module. Set up the functions to be used and the necessary parameters. Table 5-52 shows the initial values.

Table 5-51 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	SENSOR_CFG_ENCD_TIMER_FREQ	Set the frequency of the timer for captured encoder signals.
	SENSOR_CFG_ENCD_PPR	Set the number of pulses for the encoder to be used [p/r].
	SENSOR_CFG_ENCD_RESOLUTION_MULTIPL	Set the multiplication factor of encoder signals.
	SENSOR_CFG_ENCD_FUNC_CNT_GET	Set the function that acquires the count value of the timer that uses the input capture to acquire encoder signals.
	SENSOR_CFG_ENCD_FUNC_CNT_SET	Set the function that sets the count value of the timer that uses the input capture to acquire encoder signals.
	SENSOR_CFG_ENCD_FUNC_INT_ENABLE	Set the function that enables input capture interrupts of the encoder.
	SENSOR_CFG_ENCD_FUNC_INT_DISABLE	Set the function that disables input capture interrupts of the encoder.
	SENSOR_CFG_ENCD_FUNC_SPD_TIMER_START	Set the function that starts the counting of the free-run timer for speed calculation.
	SENSOR_CFG_ENCD_FUNC_SPD_TIMER_CNT_GET	Set the function that acquires the count value of the free-run timer for speed calculation.
	SENSOR_CFG_HALL_FUNC_SIGNAL_GET	Set the function that acquires Hall sensor signals.
	SENSOR_CFG_HALL_FUNC_INT_U_START	Set the function that enables U-phase interrupts of the Hall sensor.
	SENSOR_CFG_HALL_FUNC_INT_V_START	Set the function that enables V-phase interrupts of the Hall sensor.
	SENSOR_CFG_HALL_FUNC_INT_W_START	Set the function that enables W-phase interrupts of the Hall sensor.
	SENSOR_CFG_HALL_FUNC_INT_U_STOP	Set the function that disables U-phase interrupts of the Hall sensor.
	SENSOR_CFG_HALL_FUNC_INT_V_STOP	Set the function that disables V-phase interrupts of the Hall sensor.
	SENSOR_CFG_HALL_FUNC_INT_W_STOP	Set the function that disables W-phase interrupts of the Hall sensor.

Table 5-52 List of initial values for configurations

Macro name	Settings		
	RX26T RAM64KB Version	**RX26T RAM48KB Version**	**RX14T**
SENSOR_CFG_ENCD_TIMER_FREQ	120	60.0f/CMTW0_PCLK_COUNTER_DIVISION	48.0
SENSOR_CFG_ENCD_PPR	1000		16
SENSOR_CFG_ENCD_RESOLUTION_MULTIPL	4		
SENSOR_CFG_ENCD_FUNC_CNT_GET	MTU1*9	GPT5*9	MTU1*9
SENSOR_CFG_ENCD_FUNC_CNT_SET	MTU1*10	GPT5*10	MTU1*10
SENSOR_CFG_ENCD_FUNC_INT_ENABLE	R_Config_MTU0_InterruptEnable		
SENSOR_CFG_ENCD_FUNC_INT_DISABLE	R_Config_MTU0_InterruptDisable		
SENSOR_CFG_ENCD_FUNC_SPD_TIMER_START	GPT3*1	CMTW0*1	GPT0*1
SENSOR_CFG_ENCD_FUNC_SPD_TIMER_CNT_GET	GPT3*2	CMTW0*2	GPT0*2
SENSOR_CFG_HALL_FUNC_SIGNAL_GET	R_Config_PORT_HallSignalGet		
SENSOR_CFG_HALL_FUNC_INT_U_START	IRQ7*3	IRQ1*3	IRQ0*3
SENSOR_CFG_HALL_FUNC_INT_V_START	IRQ15*4	IRQ2*4	IRQ1*4
SENSOR_CFG_HALL_FUNC_INT_W_START	IRQ4*5	IRQ0*5	IRQ6*5
SENSOR_CFG_HALL_FUNC_INT_U_STOP	IRQ7*6	IRQ1*6	IRQ0*6
SENSOR_CFG_HALL_FUNC_INT_V_STOP	IRQ15*7	IRQ2*7	IRQ1*7
SENSOR_CFG_HALL_FUNC_INT_W_STOP	IRQ4*8	IRQ0*8	IRQ6*8

In Table 5-52, each peripheral name indicated with a footnote number replaces the xxxx portion of the corresponding function name in the following list of function names:

Note: 1. R_Config_xxxx_SpeedCalcTimerStart
2. R_Config_xxxx_TcntGet
3. R_Config_ICU_xxxx_Start
4. R_Config_ICU_xxxx_Start
5. R_Config_ICU_xxxx_Start
6. R_Config_ICU_xxxx_Stop
7. R_Config_ICU_xxxx_Stop
8. R_Config_ICU_xxxx_Stop
9. R_Config_xxxx_TcntGet
10. R_Config_xxxx_TcntSet

5.12.6 Structure and variable information

Table 5-53 lists the structures and variables for the sensor module.

Table 5-53 List of structures and variables

Structure	Variable	Description
st_sensor_t Structure for the sensor module	u1_ctrl_loop_mode	Control mode information
	f4_ctrl_period	Control interval information [s]
	st_ec	Structure for the encoder
	st_ehc	Structure for the encoder during high-speed rotation
	st_hc	Structure for the Hall sensor
	st_motor	Structure for motor parameters
	*EncdCntDataGet	Pointer to the function that acquires the count value of the timer that uses the input capture to acquire encoder signals
	*EncdCntDataSet	Pointer to the function that sets the count value of the timer that uses the input capture to acquire encoder signals
	*EncdInterruptEnable	Pointer to the function that enables input capture interrupts of the encoder
	*EncdInterruptDisable	Pointer to the function that disables input capture interrupts of the encoder
	*EncdTimerStart	Pointer to the function that starts the counting of the free-run timer for speed calculation
	*EncdTimerGet	Pointer to the function that acquires the count value of the free-run timer for speed calculation
	*HallSignalGet	Pointer to the function that acquires Hall sensor signals
	*HallUEnable	Pointer to the function that enables U-phase interrupts of the Hall sensor
	*HallVEnable	Pointer to the function that enables V-phase interrupts of the Hall sensor
	*HallWEnable	Pointer to the function that enables W-phase interrupts of the Hall sensor
	*HallUDisable	Pointer to the function that disables U-phase interrupts of the Hall sensor
	*HallVDisable	Pointer to the function that disables V-phase interrupts of the Hall sensor
	*HallWDisable	Pointer to the function that disables W-phase interrupts of the Hall sensor
st_encoder_t Structure for the encoder during low-speed rotation	u2_encd_pre_phase_cnt;	Phase count value of the encoder (used at switchover between low speed and high speed)
	u4_encd_timer_cap_tcnt;	Count value of the input capture of the encoder
	u4_encd_timer_cap_pre_tcnt;	Previous count value of the input capture of the encoder
	u2_encd_timer_cnt_num;	Number of the buffer for speed calculation at low speed

Structure	Variable	Description
st_encoder_t Structure for the encoder during low-speed rotation	u2_encd_cpr_mech;	Number of pulses per rotation [p/r]
	u4_encd_timer_cnt_buff;	Buffer for speed calculation at low speed
	u4_encd_pulse_width;	Pulse interval of the encoder for speed detection
	u4_encd_pulse_width_buff;	Buffer for storing the pulse interval of the encoder
	u4_encd_pulse_width_sum;	Total pulse interval of the encoder. This variable is used to detect speed 0.
	s4_encd_angle_cnt;	Counter value of the encoder for the position. This variable is used to integrate the differentials of counter values at intervals
	f4_encd_angle_diff;	Angle information for one pulse width of the encoder
	f4_encd_cpr_mech_inv;	Inverse number of the number of pulses per motor rotation
	f4_encd_speed_pre_rad;	Previous speed [rad/s]
	f4_encd_speed_rad;	Speed (during low-speed rotation) [rad/s]
	f4_encd_pos_rad;	Position (during low-speed rotation) [rad]
st_encoder_highspeed_t Structure for the encoder during high-speed rotation	u1_encd_pos_speed_calc_mode;	Selection of the position and speed detection method of the encoder
	u1_encd_interrupt_on_flag;	Interrupt flag of the encoder
	u1_encd_pos_speed_calc_cnt;	Count value for position and speed calculation of the encoder
	u2_encd_hs_pre_phase_cnt;	Count value of the encoder at the previous interval
	s4_encd_hs_angle_cnt;	Position information count value of the encoder
	f4_encd_hs_pos_rad;	Position (during high-speed rotation) [rad]
	f4_encd_hs_pos_pre_rad;	Rotor position information at the previous interval [rad]
	f4_encd_hs_speed_rad;	Speed (during high-speed rotation) [rad/s]
	f4_encd_hs_speed_pre_rad;	Speed at the previous interval [rad/s]
	f4_encd_hs_sw_speed_rad;	Speed threshold between high-speed and low-speed modes [rad/s]
	f4_encd_hs_sw_speed_margin_rad;	Margin around the threshold between high-speed and low-speed modes [rad/s]
st_hall_t Structure for the Hall sensor	u1_hall_signal;	Input value of the Hall signal
	u1_hall_pre_signal;	Previous input value of the Hall signal
	u1_hall_interrupt_flg;	Hall interrupt flag
	f4_hall_angle_rad;	Hall angle information [rad]

Structure	Variable	Description
st_sensor_encoder_cfg_t Structure for setting the parameters for controlling the encoder module	u1_ctrl_loop_mode;	Control mode (position and speed)
	u2_hs_change_speed_rpm;	Speed threshold between high-speed and low-speed modes [rpm]
	u2_hs_change_margin_rpm;	Margin around the threshold between high-speed and low-speed modes [rpm]
	u2_encd_cpr;	Number of pulses per rotation [p/r]
	f4_ctrl_period;	Control interval [s]
	st_motor	Structure for motor parameters
st_sensor_encoder_output_t Structure for encoder output	f4_speed_rad;	Speed [rad/s]
	f4_pos_rad;	Position [rad]
st_sensor_hall_output_t Structure for Hall sensor output	f4_hall_angle_rad;	Hall angle [rad]
st_encoder_driver_cfg_t Structure for setting the sensor module function pointer	*EncdCntDataGet	Pointer to the function that acquires the count value of the timer that uses the input capture to acquire encoder signals
	*EncdCntDataSet	Pointer to the function that sets the count value of the timer that uses the input capture to acquire encoder signals
	*EncdInterruptEnable	Pointer to the function that enables input capture interrupts of the encoder
	*EncdInterruptDisable	Pointer to the function that disables input capture interrupts of the encoder
	*EncdTimerStart	Pointer to the function that starts the counting of the free-run timer for speed calculation
	*EncdTimerGet	Pointer to the function that acquires the count value of the free-run timer for speed calculation
	*HallSignalGet	Pointer to the function that acquires Hall sensor signals
	*HallUEnable	Pointer to the function that enables U-phase interrupts of the Hall sensor
	*HallVEnable	Pointer to the function that enables V-phase interrupts of the Hall sensor
	*HallWEnable	Pointer to the function that enables W-phase interrupts of the Hall sensor
	*HallUDisable	Pointer to the function that disables U-phase interrupts of the Hall sensor
	*HallVDisable	Pointer to the function that disables V-phase interrupts of the Hall sensor
	*HallWDisable	Pointer to the function that disables W-phase interrupts of the Hall sensor

5.12.7 Macro definition

Table 5-54 lists the macros for the sensor module.

Table 5-54 List of macros

File name	Macro name	Defined value	Remarks
r_motor_sensor_api.h	SENSOR_ENCD_ANGLE_ADJ_TIME	RX26T: 512 RX14T: 256	Adjustment value for the rotor pull-in time at startup by forced excitation (position/speed-interrupt-interval × adjustment-value = pull-in time)
	SENSOR_ENCD_CNTAVG	4	Number of pulse intervals required to average the speed
	SENSOR_HALL_EDGE_ERROR	1	Hall signal detection error
	SENSOR_ENCD_NUMB_OF_TIME	2	Maximum number of encoder interrupts that can be processed per control interval
	SENSOR_ENCD_HS_CHANGE_RPM	(SENSOR_ENCD_NUMB_OF_TIME * 60)/ (MOTOR_COMMON_CTRL_PERIOD * MOTOR_COMMON_SENSOR_ENCD_CPR)	Speed at which the basis of speed detection changes from the pulse interval to the number of pulses within a certain interval [rpm/s]
	SENSOR_ENCD_HS_CHANGE_RAD	(SENSOR_ENCD_HS_CHANGE_RPM * MTR_RPM2RAD)	Speed at which the basis of speed detection changes from the pulse interval to the number of pulses within a certain interval [rad]
	SENSOR_ENCD_HS_CHANGE_MARGIN_RPM	150	Margin of the speed at which the basis of speed detection changes from the pulse interval to the number of pulses within a certain interval [rpm]
	SENSOR_ENCD_HS_CHANGE_MARGIN_RAD	SENSOR_ENCD_HS_CHANGE_MARGIN_RPM * MTR_RPM2RAD	Margin of the speed at which the basis of speed detection changes from the pulse interval to the number of pulses within a certain interval [rad/s]
	SENSOR_ENCD_LOOP_POSITION	0	Perform the operations required for position control
	SENSOR_ENCD_LOOP_SPEED	1	Perform the operations required for speed control

5.12.8 Adjustment and configuration of parameters

The initial values of sensor module parameters can be specified with the configuration information (r_motor_module_cfg.h). The specified configurations are applied when the system starts. For details about the items to be set, see 5.12.5.

5.12.9 Switchover of the method for calculating the position and speed

A general method to calculate the position and speed from encoder signals is to count signal edges. However, if you use an encoder with a low resolution, low speed cannot be calculated accurately due to the large ratio of encoder pulse interval to control interval. For this reason, a method that uses a free-run timer to measure pulse intervals is implemented for calculation at low speed. In this method, interrupts are generated by using encoder signals to calculate the position and speed.

However, if this method is used with a high rotation speed or high-resolution encoder, a large number of interrupts are generated within a control interval. This may lead to excessive resource usage in interrupt generation processing and cause control breakdown.

To prevent this, the calculation method switches between calculation based on encoder signal interrupts and calculation based on carrier wave interrupts at a certain speed. Figure 5-35 shows an overview of this switchover. As shown in this figure, the speed calculation method changes from calculation based on encoder interrupts to calculation based on current control interrupts when the speed becomes high.

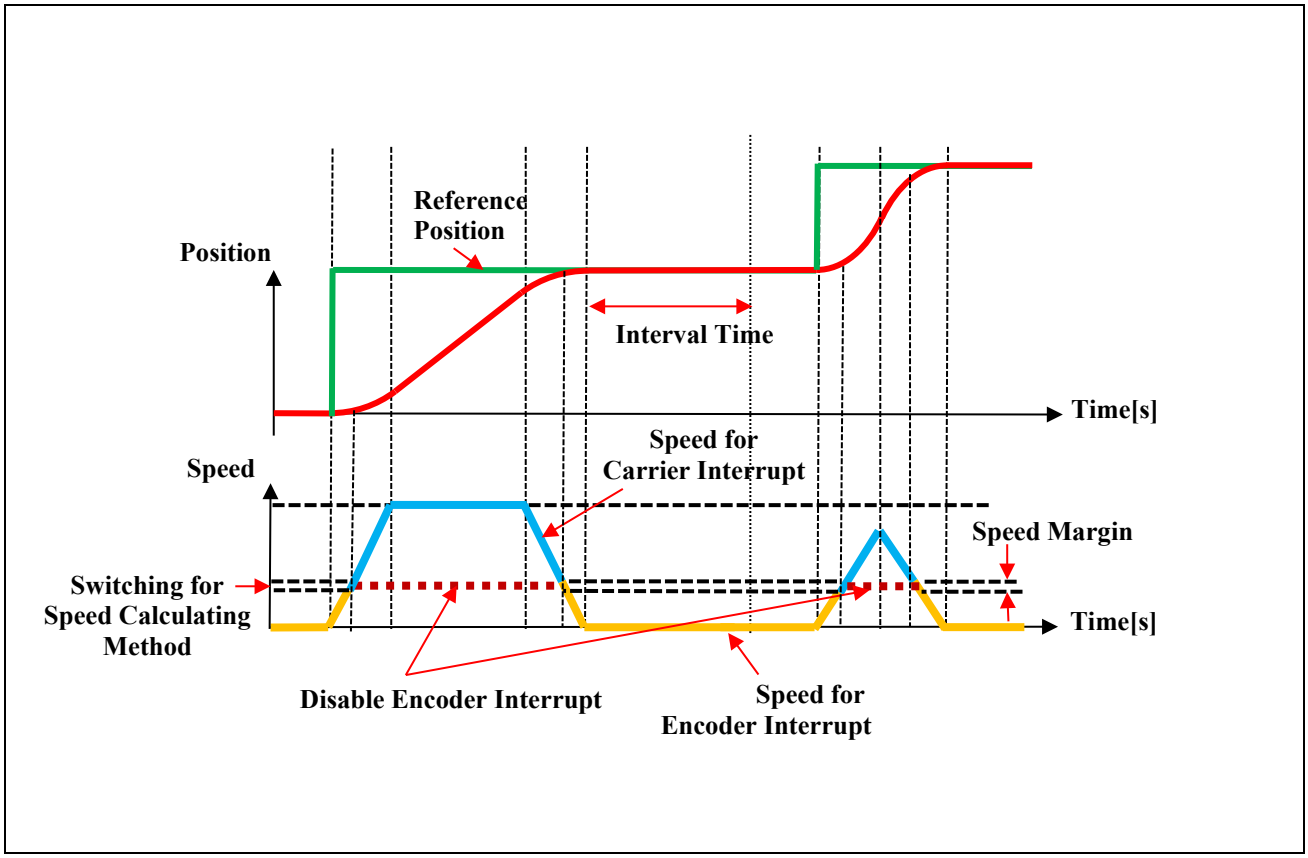


Figure 5-35 Switchover of the method for calculating the position and speed (example)

5.12.10 Method for calculating the position and speed by using a speed sensor

(a) Speed calculation using an encoder at low speed

Speed calculation using an encoder at low speed is performed as shown in Figure 5-36.

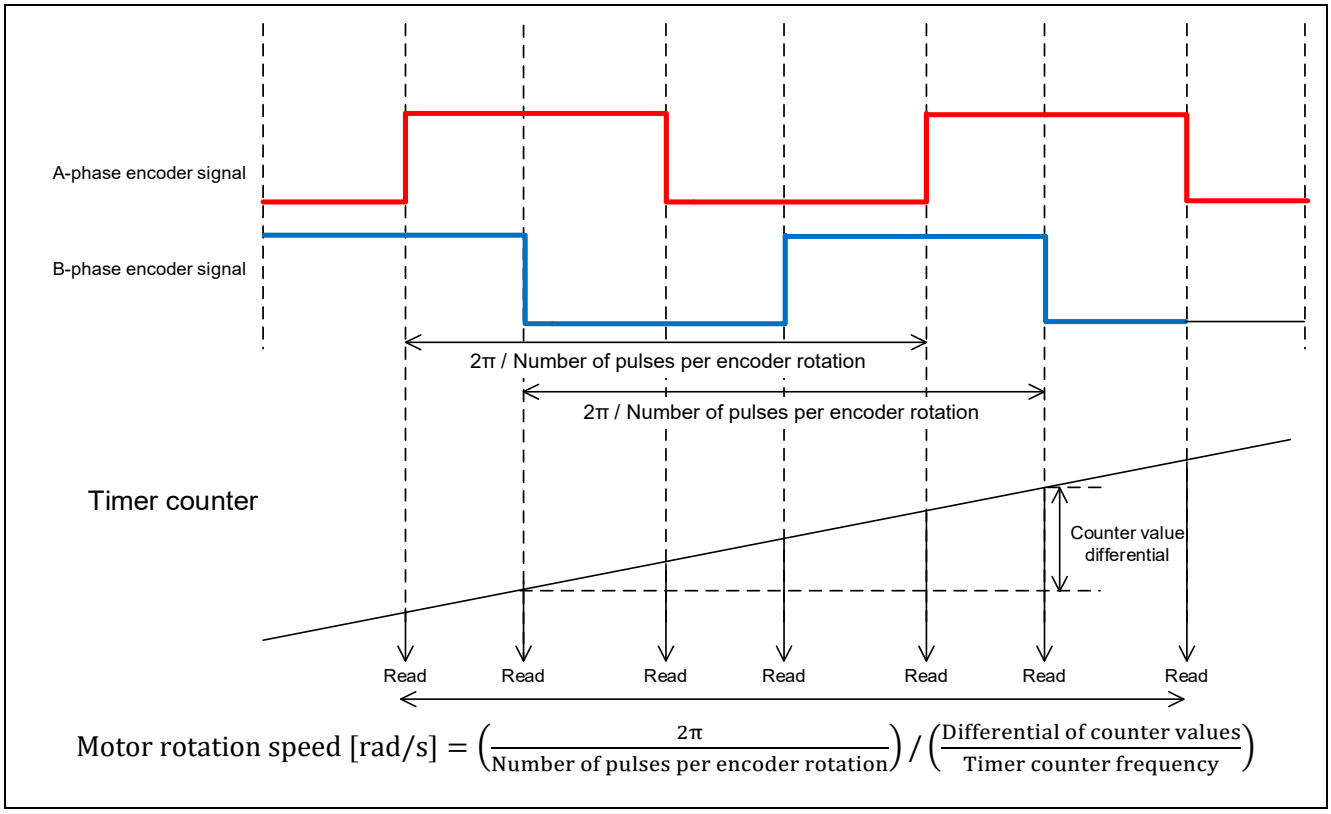
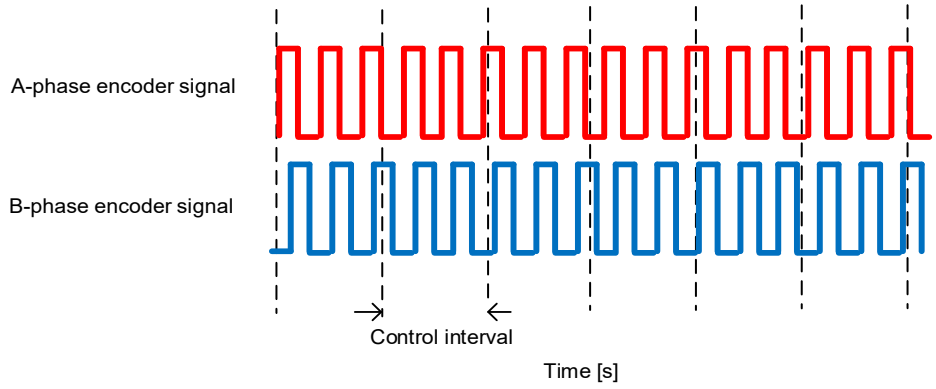


Figure 5-36 Speed calculation using an encoder at low speed

(b) Speed calculation using an encoder at high speed

Speed calculation using an encoder at high speed is performed as shown in Figure 5-37.



$$\text{Motor rotation speed [rad/s]} = \left(\frac{2\pi}{\text{Number of pulses per encoder rotation}}\right) \times \left(\frac{\text{Number of pulses input within a control interval}}{\text{Control interval}}\right)$$

Figure 5-37 Speed calculation using an encoder at high speed

5.13 Driver module

The driver module works as an interface between the manager module, which corresponds to the middleware of the sample software, and Smart Configurator, which is required to access the MCU peripherals. Configuring the driver module appropriately allows you to use MCU function allocation and the differentials of the board to be used without modifying the motor module.

5.13.1 Functionality

Table 5-55 lists the functions of the driver module.

Table 5-55 List of functions of the driver module

Function	Description
Acquisition of the A/D conversion value	Acquires AD values such as the phase current and inverter board bus voltage via a Smart Configurator function.
PWM duty setting	Sets the PWM duty value that is to be output to U-, V-, and W-phases via a Smart Configurator function.
PWM start/stop	Controls whether to start or stop PWM output via a Smart Configurator function.

5.13.2 Module configuration diagram

Figure 5-38 shows the driver module configuration diagram.

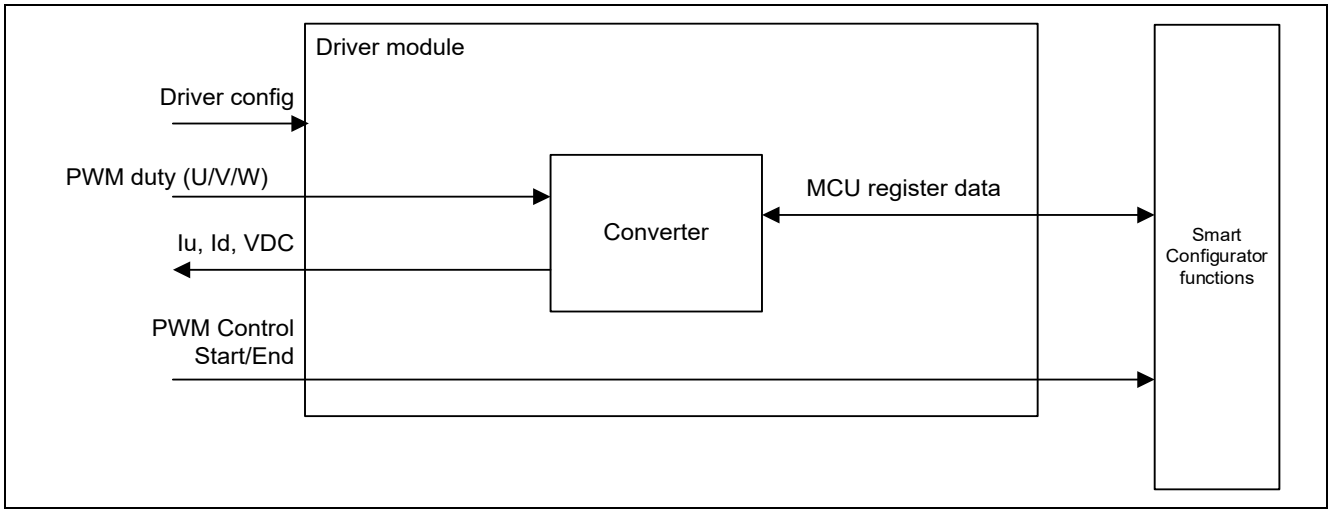


Figure 5-38 Driver module configuration diagram

5.13.3 API function

Table 5-56 lists and describes the API functions for the driver module.

Table 5-56 List of API functions

API function	Description
R_MOTOR_DRIVER_Open	Generates an instance of the driver module.
R_MOTOR_DRIVER_Close	Places the module in a reset state.
R_MOTOR_DRIVER_ParameterUpdate	Inputs the variable information that is to be used inside the module.
R_MOTOR_DRIVER_BldcAnalogGet	Acquires the A/D conversion results.
R_MOTOR_DRIVER_BldcDutySet	Sets the PWM duty.
R_MOTOR_DRIVER_PWMControlStop	Stops PWM control.
R_MOTOR_DRIVER_PWMControlStart	Starts PWM control.

5.13.4 Configurations

Table 5-57 lists the configurations for the driver module. Set up the functions to be used and the necessary parameters. Table 5-58 shows the initial values.

Table 5-57 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	DRIVER_CFG_FUNC_PWM_OUTPUT_START	Sets the function that enables PWM output.
	DRIVER_CFG_FUNC_PWM_OUTPUT_STOP	Sets the function that disables PWM output.
	DRIVER_CFG_FUNC_ADC_DATA_GET	Sets the function that acquires the A/D conversion results.
	DRIVER_CFG_FUNC_DUTY_SET	Sets the function that sets the duty cycle.
r_motor_inverter_cfg.h	INVERTER_CFG_ADC_REF_VOLTAGE	Sets the reference voltage for A/D conversion.
r_motor_module_cfg.h	MOTOR_MCU_CFG_ADC_OFFSET	Sets the AD offset value.

Table 5-58 List of initial values for configurations

Macro name	Settings
DRIVER_CFG_FUNC_PWM_OUTPUT_START	R_Config_MOTOR_StartTimerCtrl (Smart Configurator function) *1
DRIVER_CFG_FUNC_PWM_OUTPUT_STOP	R_Config_MOTOR_StopTimerCtrl (Smart Configurator function) *1
DRIVER_CFG_FUNC_ADC_DATA_GET	RX26T: R_Config_xxx_AdcGetConvVal RX14T: R_Config_xxx_AdcGetConvVal_MultiUnit (Smart Configurator function) *1
DRIVER_CFG_FUNC_DUTY_SET	R_Config_MOTOR_UpdDuty (Smart Configurator function) *1
INVERTER_CFG_ADC_REF_VOLTAGE	5.0f
MOTOR_MCU_CFG_ADC_OFFSET	0x7FF

Note: 1. For details about the functions shown in the "Set value" column, see 5.14 Smart Configurator settings.

5.13.5 Structure and variable information

Table 5-59 lists the structures that are used for the driver module. In the driver module, the structure for the driver module (g_st_driver) is defined by securing an instance of the module from the API.

Table 5-59 List of structures and variables

Structure	Variable	Description
st_motor_driver_t Structure for the driver module	*ADCDataGet	Pointer to the Smart Configurator function (This variable sets the function that acquires the results of A/D conversion.)
	*BLDCDutySet	Pointer to the Smart Configurator function (This variable sets the function that enables PWM output.)
	*PWMOutputStop	Pointer to the Smart Configurator function (This variable sets the function that disables PWM output.)
	*PWMOutputStart	Pointer to the Smart Configurator function (This variable sets the function that sets the duty cycle.)
	f4_ad_crnt_per_digit	Scale for A/D conversion of the current
	f4_ad_vdc_per_digit	Scale for A/D conversion of the voltage
	f4_pwm_period_cnt	Count value for one interval of the PWM counter (information for the duty setting)
	f4_pwm_dead_time_cnt	Count value for the dead time (information for the duty setting)
st_motor_driver_cfg_t Structure for setting the parameters for controlling the driver module	*ADCDataGet	Pointer to the Smart Configurator function
	*BLDCDutySet	Pointer to the Smart Configurator function
	*PWMOutputStop	Pointer to the Smart Configurator function
	*PWMOutputStart	Pointer to the Smart Configurator function
	f4_shunt_ohm	Shunt resistance value [ohm] (for calculation of f4_ad_crnt_per_digit)
	f4_volt_gain	Voltage conversion gain coefficient (for calculation of f4_ad_vdc_per_digit)
	f4_crnt_amp_gain	Current conversion gain coefficient (for calculation of f4_ad_crnt_per_digit)
	f4_pwm_period_cnt	Count value for one interval of the PWM counter (information for the duty setting)
	f4_pwm_dead_time_cnt	Count value for the dead time (information for the duty setting)

5.13.6 Macro definition

Table 5-60 lists the macros for the driver module.

Table 5-60 List of macros

File name	Macro name	Defined value	Remarks
r_motor_driver.c	MOTOR_DRIVER_PRV_ADC_REF_VOLTAGE	INVERTER_CFG_ADC_REF_VOLTAGE (See Table 5-57.)	Reference voltage [V]

5.13.7 Adjustment and configuration of parameters

(a) Setting the parameters for controlling the driver module

In the driver module, parameters that are input from the control parameter configuration (R_MOTOR_DRIVER_ParameterUpdate) are used to associate the motor module and Smart Configurator and to convert data. The parameters are input by using st_speed_cfg_t (the structure for setting the parameters for controlling the driver module). In the sample program, the information that is defined as configurations is used as the parameter settings. Table 5-61 shows the settings.

Table 5-61 Example of settings specified in the sample program

Variable name	Macro name	File name
*ADCDataGet	DRIVER_CFG_FUNC_ADC_DATA_GET	See Table 5-57.
*BLDCDutySet	DRIVER_CFG_FUNC_DUTY_SET	
*PWMOutputStop	DRIVER_CFG_FUNC_PWM_OUTPUT_START	
*PWMOutputStart	DRIVER_CFG_FUNC_PWM_OUTPUT_STOP	
f4_shunt_ohm	INVERTER_CFG_SHUNT_RESIST	r_motor_inverter_cfg.h
f4_volt_gain	INVERTER_CFG_VOLTAGE_GAIN	
f4_crnt_amp_gain	INVERTER_CFG_CURRENT_AMP_GAIN	
f4_pwm_period_cnt	MOTOR_COMMON_CARRIER_SET_BASE	r_motor_module_cfg.h
f4_pwm_dead_time_cnt	MOTOR_COMMON_DEADTIME_SET	

5.14 Smart Configurator settings

In the sample program, Smart Configurator is used to create a project. This section describes the components used and the functions added to the user area.

5.14.1 Clock settings

Table 5-62 and Table 5-63 shows the clock settings.

Table 5-62 MCU clock settings

Clock	Frequency	
	RX26T RAM64KB Version	RX26T RAM48KB Version
Main clock	10MHz	
System clock (ICLK)	120MHz	
Peripheral module clock (PCLKA)	120MHz	
Peripheral module clocks (PCLKB, PCLKC, and PCLKD)	60MHz / 120MHz / 60MHz	
Flash IF clock (FCLK)	60MHz	
IWDTCLK	120kHz	

Table 5-63 MCU clock settings (RX14T)

Clock	Frequency
	RX14T
Main clock	48MHz
System clock (ICLK)	48MHz
Peripheral module clocks (PCLKB/PCLKD)	48MHz
Flash IF clock (FCLK)	48MHz
IWDTCLK	15kHz

5.14.2 Component settings

Table 5-64 lists the components used in the each MCU and the functions allocated to the components.

Table 5-64 Smart Configurator components and their functions

Allocated function	**Component**		
	RX26T RAM64KB Version	**RX26T RAM48KB Version**	**RX14T**
Hall interrupt processing	Config_ICU		
3-phase PWM output, A/D conversion processing (inverter bus voltage detection, current detection)	Config_MOTOR		
A/D conversion processing (command voltage detection for the board UI)	Config_S12AD2		Config_S12AD1
Setting of the port to be used	Config_PORT		
Position and speed control interrupt timer	Config_CMT0		
Free-run timer for speed measurement	Config_GPT3	Config_CMTW0	Config_GPT0
Independent watchdog timer	Config_IWDT		
Phase counter for the encoder	Config_MTU0		
Event Link (for Phase counter)	---	Config_ELC	---
Phase counter for the encoder	Config_MTU1	Config_GPT5	Config_MTU1
Overcurrent detection	Config_POE		

5.14.3 Interrupts

Table 5-65 , Table 5-66 and Table 5-67 shows interrupt information for MCUs that use motor components.

Table 5-65 List of interrupts (RX26T RAM64KB Version)

Component	Interrupt function	Description
Config_ICU	r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1	Interrupts of the Hall sensor Interrupt level: 14 Multiple interrupt: Disabled
Config_MOTOR	r_Config_MOTOR_ad_interrupt	A/D conversion end interrupt Interrupt level: 12 Multiple interrupt: Enabled
Config_S12AD2	None	None
Config_PORT	None	None
Config_CMT0	r_Config_CMT0_cmi0_interrupt	Position and speed control interrupt Interrupt level: 11 Multiple interrupt: Enabled
Config_GPT3	None	None
Config_IWDT	None	None
Config_MTU0	r_Config_MTU0_tgib0_interrupt	Encoder interrupt Interrupt level: 13 Multiple interrupt: Disabled
Config_MTU1	None	None
Config_POE	r_Config_POE_oei1_interrupt	Hardware overcurrent interrupt / Short-circuited outputs interrupt Interrupt level: 15 Multiple interrupt: Disabled

Note: 1. “xx” is the setting assigned to each MCU. See Table 5-52 for details.

Table 5-66 List of interrupts (RX26T RAM48KB Version)

Component	Interrupt function	Description
Config_ICU	r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1	Interrupts of the Hall sensor Interrupt level: 14 Multiple interrupt: Disabled
Config_MOTOR	r_Config_MOTOR_ad_interrupt	A/D conversion end interrupt Interrupt level: 12 Multiple interrupt: Enabled
Config_S12AD2	None	None
Config_PORT	None	None
Config_CMT0	r_Config_CMT0_cmi0_interrupt	Position and speed control interrupt Interrupt level: 11 Multiple interrupt: Enabled
Config_GPT3	None	None
Config_IWDT	None	None
Config_MTU0	r_Config_MTU0_tgib0_interrupt	Encoder interrupt Interrupt level: 13 Multiple interrupt: Disabled
Config_MTU1	None	None
Config_POE	r_Config_POE_oei1_interrupt	Hardware overcurrent interrupt Interrupt level: 15 Multiple interrupt: Disabled
	r_Config_POE_oei2_interrupt	Short-circuited outputs interrupt Interrupt level: 15 Multiple interrupt: Disabled

Note: 1. “xx” is the setting assigned to each MCU. See Table 5-52 for details.

Table 5-67 List of interrupts (RX14T)

Component	Interrupt function	Description
Config_ICU	r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1	Interrupts of the Hall sensor Interrupt level: 14 Multiple interrupt: Disabled
Config_MOTOR	r_Config_MOTOR_ad_interrupt	A/D conversion end interrupt Interrupt level: 12 Multiple interrupt: Enabled
Config_S12AD1	None	None
Config_PORT	None	None
Config_CMT0	r_Config_CMT0_cmi0_interrupt	Position and speed control interrupt Interrupt level: 11 Multiple interrupt: Enabled
Config_GPT0	None	None
Config_IWDT	None	None
Config_MTU0	r_Config_MTU0_tgib0_interrupt	Encoder interrupt Interrupt level: 13 Multiple interrupt: Disabled
Config_MTU1	None	None
Config_POE	r_Config_POE_oei4_interrupt	Hardware overcurrent interrupt Interrupt level: 15 Multiple interrupt: Disabled

5.14.4 Details of user codes

Table 5-68, Table 5-69 and Table 5-70 lists the functions that are created in the user code area.

Table 5-68 List of functions in the user area (RX26T RAM64KB Version)

Component	Function	Description
Config_PORT	R_Config_PORT_GetSW1	Acquires the status of SW1.
	R_Config_PORT_GetSW2	Acquires the status of SW2.
	R_Config_PORT_Led1_on	Turns on LED1.
	R_Config_PORT_Led2_on	Turns on LED2.
	R_Config_PORT_Led1_off	Turns off LED1.
	R_Config_PORT_Led2_off	Turns off LED2.
Config_GPT3	R_Config_GPT3_TcntGet	Acquires the Timer counter value.
	R_Config_GPT3_SpeedCalcTimerStart	Starts the free-run timer for speed measurement.
Config_MTU0	R_Config_MTU0_InterruptEnable	Enables encoder interrupts.
	R_Config_MTU0_InterruptDisable	Disables encoder interrupts.
Config_MTU1	R_Config_MTU1_TcntSet	Sets the MTU1 timer counter.
	R_Config_MTU1_TcntGet	Reads the value from the MTU1 timer counter.

Table 5-69 List of functions in the user area (RX26T RAM48KB Version)

Component	Function	Description
Config_PORT	R_Config_PORT_GetSW1	Acquires the status of SW1.
	R_Config_PORT_GetSW2	Acquires the status of SW2.
	R_Config_PORT_Led1_on	Turns on LED1.
	R_Config_PORT_Led2_on	Turns on LED2.
	R_Config_PORT_Led1_off	Turns off LED1.
	R_Config_PORT_Led2_off	Turns off LED2.
Config_CMTW0	R_Config_CMTW0_TcntGet	Acquires the Timer counter value.
	R_Config_CMTW0_SpeedCalcTimerStart	Starts the free-run timer for speed measurement.
Config_MTU0	R_Config_MTU0_InterruptEnable	Enables encoder interrupts.
	R_Config_MTU0_InterruptDisable	Disables encoder interrupts.
Config_GPT5	R_Config_GPT5_TcntSet	Sets the GPT5 timer counter.
	R_Config_GPT5_TcntGet	Reads the value from the GPT5 timer counter.

Table 5-70 List of functions in the user area (RX14T)

Component	Function	Description
Config_PORT	R_Config_PORT_GetSW1	Acquires the status of SW1.
	R_Config_PORT_GetSW2	Acquires the status of SW2.
	R_Config_PORT_Led1_on	Turns on LED1.
	R_Config_PORT_Led2_on	Turns on LED2.
	R_Config_PORT_Led1_off	Turns off LED1.
	R_Config_PORT_Led2_off	Turns off LED2.
	R_Config_PORT_Led3_on	Turns on LED3.
	R_Config_PORT_Led3_off	Turns off LED3.
Config_GPT0	R_Config_GPT0_TcntGet	Acquires the Timer counter value.
	R_Config_GPT0_SpeedCalcTimerStart	Starts the free-run timer for speed measurement.
Config_MTU0	R_Config_MTU0_InterruptEnable	Enables encoder interrupts.
	R_Config_MTU0_InterruptDisable	Disables encoder interrupts.
Config_MTU1	R_Config_MTU1_TcntSet	Sets the MTU1 timer counter.
	R_Config_MTU1_TcntGet	Reads the value from the MTU1 timer counter.

5.14.5 Pin settings

Table 5-71, Table 5-72 and Table 5-73 shows the pin interface information.

Table 5-71 Pin interface (RX26T RAM64KB Version)

Function	Pin Name
Measurement of the inverter bus voltage	P43 / AN003
Position/speed command value (analog value)	P50 / AN204
Start/stop toggle switch	P23
Error reset push switch	P22
Controlling LED1	P21
Controlling LED2	P20
Measurement of the U-phase current	P40 / AN000
Measurement of the W-phase current	P42 / AN002
PWM output (U_p)	P73 / MTIOC4B
PWM output (V_p)	P72 / MTIOC4A
PWM output (W_p)	P71 / MTIOC3B
PWM output (U_n)	P76 / MTIOC4D
PWM output (V_n)	P75 / MTIOC4C
PWM output (W_n)	P74 / MTIOC3D
Hall U-phase input	P30 / IRQ7
Hall V-phase input	P27 / IRQ15
Hall W-phase input	P24 / IRQ4
Encoder A-phase input	P33 / MTCLKA
Encoder B-phase input	P32 / MTCLKB
PWM emergency stop input when an overcurrent is detected	P70 / POE0#

Table 5-72 Pin interface (RX26T RAM48KB Version)

Function	Pin Name
Measurement of the inverter bus voltage	P43 / AN003
Position/speed command value (analog value)	P47 / AN206
Start/stop toggle switch	P21
Error reset push switch	P20
Controlling LED1	P65
Controlling LED2	PB5
Measurement of the U-phase current	P40 / AN000
Measurement of the W-phase current	P42 / AN002
PWM output (U_p)	P71 / MTIOC3B
PWM output (V_p)	P72 / MTIOC4A
PWM output (W_p)	P73 / MTIOC4B
PWM output (U_n)	P74 / MTIOC3D
PWM output (V_n)	P75 / MTIOC4C
PWM output (W_n)	P76 / MTIOC4D
Hall U-phase input	P11 / IRQ1
Hall V-phase input	P00 / IRQ2
Hall W-phase input	PE2 / IRQ0
Encoder A-phase input	P94 / MTCLKA
Encoder B-phase input	P91 / MTCLKB
PWM emergency stop input when an overcurrent is detected	P96 / POE4#

Table 5-73 Pin interface (RX14T)

Function	Pin Name
Measurement of the inverter bus voltage	P11 / AN102
Position/speed command value (analog value)	P10 / AN103
Start/stop toggle switch	PD7
Error reset push switch	PE2
Controlling LED1	PD3
Controlling LED2	PD4
Controlling LED3	PD6
Measurement of the U-phase current	P41 / AN001
Measurement of the W-phase current	P40 / AN000
PWM output (U_p)	P73 / MTIOC4B
PWM output (V_p)	P72 / MTIOC4A
PWM output (W_p)	P71 / MTIOC3B
PWM output (U_n)	P76 / MTIOC4D
PWM output (V_n)	P75 / MTIOC4C
PWM output (W_n)	P74 / MTIOC3D
Hall U-phase input	P92 / IRQ0
Hall V-phase input	P93 / IRQ1
Hall W-phase input	P90 / IRQ6
Encoder A-phase input	P33 / MTCLKA
Encoder B-phase input	P32 / MTCLKB
PWM emergency stop input when an overcurrent is detected	P30 / POE11#

5.14.6 Macro definition

Table 5-74 and Table 5-75 lists the macros for the driver module.

Table 5-74 List of macros (RX26T RAM64KB Version / RX26T RAM48KB Version)

File name	Macro name	Defined value	Remarks
Config_MOTOR .h	CG_CONFIG_MOTOR_PWM_ TIMER_FREQ	120.0f	PWM timer frequency [MHz]
	CG_CONFIG_MOTOR_CARRI ER_FREQ	20.000f	Carrier wave actual frequency [kHz]
	CG_CONFIG_MOTOR_DEADT IME	2.0f	Deadtime actual value [us]
	CG_CONFIG_MOTOR_INTR_ DECIMATION	0.0	Interrupt skipping number
	CG_MOTOR_CFG_MAX_AD_ DATA	4095.0f	Max A/D data
	CG_MOTOR_MCU_CFG_AD_ FREQ	60.0f	A/D frequency [MHz]

Table 5-75 List of macros (RX14T)

File name	Macro name	Defined value	Remarks
Config_MOTOR .h	CG_CONFIG_MOTOR_PWM_ TIMER_FREQ	48.0f	PWM timer frequency [MHz]
	CG_CONFIG_MOTOR_CARRI ER_FREQ	20.000f	Carrier wave actual frequency [kHz]
	CG_CONFIG_MOTOR_DEADT IME	1.000f	Deadtime actual value [us]
	CG_CONFIG_MOTOR_INTR_ DECIMATION	0.0	Interrupt skipping number
	CG_MOTOR_CFG_MAX_AD_ DATA	4095.0f	Max A/D data
	CG_MOTOR_MCU_CFG_AD_ FREQ	48.0f	A/D frequency [MHz]

6. Vector control algorithm

6.1 Analysis model of permanent magnet synchronous motor

The voltage equation of a permanent magnet synchronous motor with a sinusoidal magnetic flux distribution as shown in Figure 6-1 can be expressed as follows.

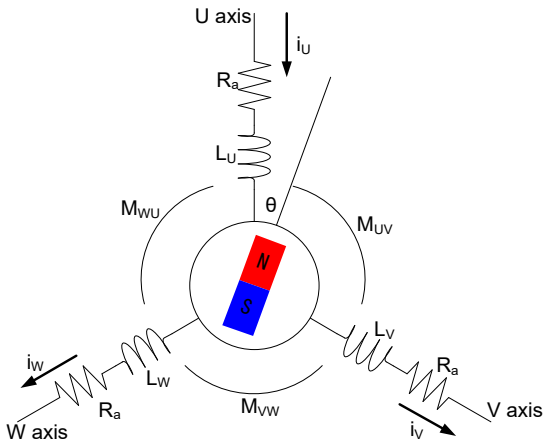


Figure 6-1 Conceptual diagram of 3-phase permanent magnet synchronous motor

$$\begin{bmatrix} v_u \\ v_v \\ v_w \end{bmatrix} = R_a \begin{bmatrix} i_u \\ i_v \\ i_w \end{bmatrix} + p \begin{bmatrix} \phi_u \\ \phi_v \\ \phi_w \end{bmatrix}$$

$$\begin{bmatrix} \phi_u \\ \phi_v \\ \phi_w \end{bmatrix} = \begin{bmatrix} L_u & M_{uv} & M_{wu} \\ M_{uv} & L_v & M_{vw} \\ M_{wu} & M_{vw} & L_w \end{bmatrix} \begin{bmatrix} i_u \\ i_v \\ i_w \end{bmatrix} + \psi \begin{bmatrix} cos\theta \\ \cos(\theta - 2\pi/3) \\ \cos(\theta + 2\pi/3) \end{bmatrix}$$

$\boldsymbol{v_u, v_v, v_w}$: Each phase armature voltage

$\boldsymbol{i_u, i_v, i_w}$: Each phase armature current

$\boldsymbol{\phi_u, \phi_v, \phi_w}$: Each phase armature interlinkage magnetic flux

$\boldsymbol{R_a}$: Each phase armature resistance

$\boldsymbol{p}$: Derivative operator

$\boldsymbol{L_u, L_v, L_w}$: Each phase self-inductance

M_{uv}, M_{vw}, M_{wu} : Mutual inductance between each phase

ψ : Maximum value of armature interlinkage magnetic flux due to permanent magnet

θ : Advance angle of permanent magnet (rotor) from U-phase

6.2 dq axis model of permanent magnet synchronous motor

In vector control, the AC 3-phase (u, v, w) coordinate system is represented by the DC 2-phase (d, q) coordinate system. The three-phase winding of the stator is converted into a two-phase winding that rotates in synchronization with the rotor of the permanent magnet, so it can be treated as two relatively stationary, electrically independent DC circuits.

In a two-phase (d, q) coordinate system, the d-axis is defined in the direction of the magnetic flux (N pole) of the rotor's permanent magnet, and the q-axis is 90 degrees forward from d. Use the following transformation matrix to get the voltage equation for a permanent magnet synchronous motor as seen from the dq coordinate system.

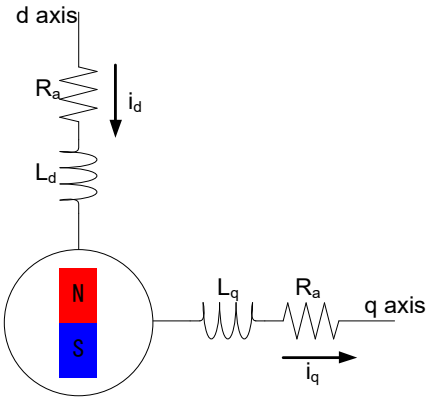


Figure 6-2 Conceptual diagram of 2-phase DC motor

$$C = \sqrt{\frac{2}{3}}\begin{bmatrix} cos\theta & \cos(\theta - 2\pi/3) & \cos(\theta + 2\pi/3) \\ -sin\theta & -\sin(\theta - 2\pi/3) & -\sin(\theta + 2\pi/3) \end{bmatrix}$$

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = C \begin{bmatrix} v_u \\ v_v \\ v_w \end{bmatrix}$$

With the above coordinate transformation, the voltage equation of the dq coordinate system can be expressed as:

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R_a + pL_d & -\omega L_q \\ \omega L_d & R_a + pL_q \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \psi_a \end{bmatrix}$$

v_d, v_q: dq axis armature voltage

i_d, i_q: dq axis armature current

R_a: Armature resistance of each phase

ω: Angular velocity

L_d, L_q: dq axis self-inductance

$L_d = l_a + \frac{3(L_a - L_{as})}{2}, L_q = l_a + \frac{3(L_a + L_{as})}{2}$

ψ_a: Effective value of armature connection magnetic flux by permanent magnet

$\psi_a = \sqrt{\frac{3}{2}}\psi$

From the above, the alternating current flowing through the stationary three-phase stator can be regarded as the direct current flowing through the two-phase stator rotating in synchronization with the permanent magnet that is the rotor.

$$T = P_n\{\psi_a i_q + (L_d - L_q) i_d i_q\}$$

T: Motor torque P_n: Pole logarithm

A motor with no difference in inductance between the d-axis and the q-axis is called a motor with no salient polarity. In this case, the reluctance torque becomes 0, so the torque increases in proportion to the q-axis current. For this reason, the q-axis current is sometimes called the torque current. On the other hand, the d-axis current is sometimes called an exciting current because it works as if the magnitude of the magnetic flux of the permanent magnet is changing by changing its magnitude.

6.3 Vector control system and controller

The block diagram of the entire position control system is shown below.

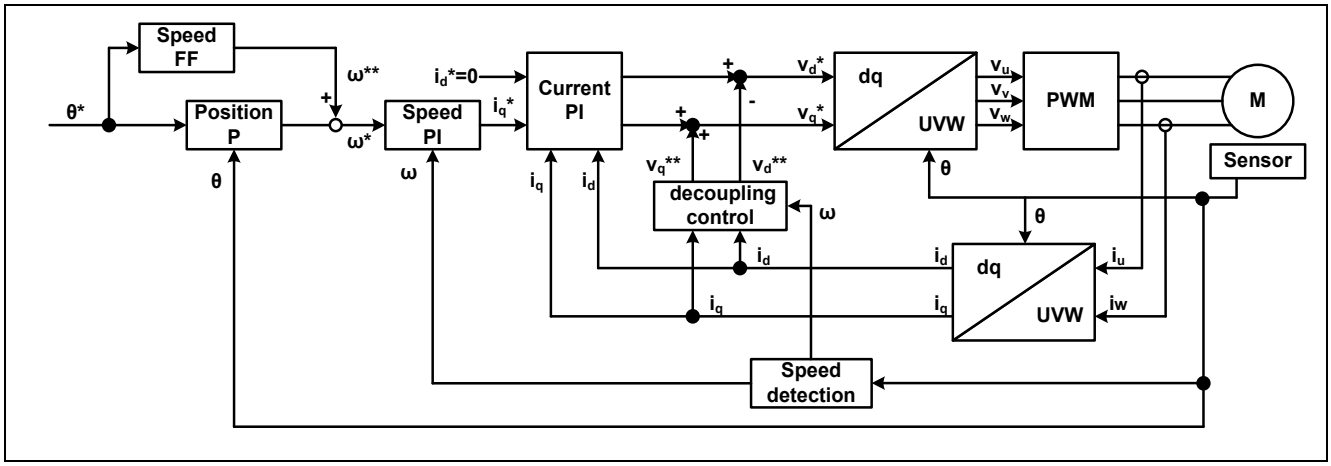


Figure 6-3 Vector control system block (position control)

As shown in Figure 6-3, the position control system consists of a position control system, a speed control system, and a current control system. Speed control systems and current control systems are realized using common PI control controllers, and position control systems are realized using P control and feedforward control. The gain of each controller must be properly designed to obtain the required control characteristics.

In the decoupling control block of the system block diagram, the induced voltage generated by the rotation of the motor v_d^{**}, v_d^{**} (see the formula below) is fed forward to the command voltage of each phase. This makes it possible to achieve high responsiveness of the speed control system and control the d-axis and q-axis independently.

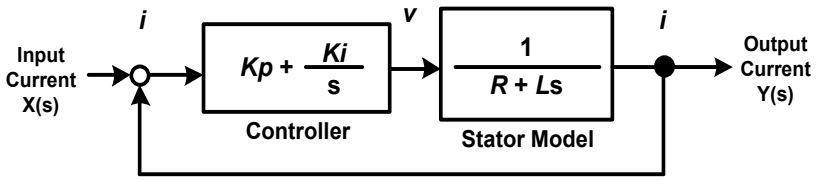
$$v_d^{**} = -\omega L_d i_q$$

$$v_q^{**} = \omega(L_d i_d + \psi_a)$$

6.3.1 Current control system design

Model the current control system from the electrical characteristics of the motor. Since the stator coil can be represented by resistance R and inductance L, the stator model of the motor can be represented by the transfer function $\frac{1}{R+Ls}$ of a typical RL series circuit.

The controller can use PI control, and the current control system can be represented by a feedback control system as shown in Figure 6-4.



R: Stator coil resistance [Ω] L: Stator coil inductance [H]

K_p: Proportional gain of current PI control K_i: Integral gain of current PI control

Figure 6-4 Current control system model

First, use the known motor stators R and L to set the PI control gain of the current control system.

The closed-loop transfer function of the current control system is obtained as follows.

$$G(s) = \frac{Y(s)}{X(s)} = \frac{\frac{K_a}{K_b}\left(1 + \frac{s}{a}\right)}{s^2 + \frac{1}{K_b}(1 + \frac{K_a}{a})s + \frac{K_a}{K_b}}$$

$$K_i = K_p a, \quad K_a = \frac{K_p a}{R}, \quad K_b = \frac{L}{R}$$

Also, the general formula for a zero quadratic lag system can be written as:

$$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}\left(1 + \frac{s}{\omega_z}\right)$$

Then, comparing the transfer function of the current control system with the general formula of the quadratic lag system with a zero point, the following relationship is obtained.

$$\frac{\omega_n^2\left(1+\frac{s}{\omega_z}\right)}{s^2+2\zeta\omega_n s+\omega_n^2} \Leftrightarrow \frac{\frac{K_a}{K_b}\left(1+\frac{s}{a}\right)}{s^2+\frac{1}{K_b}(1+\frac{K_a}{a})s+\frac{K_a}{K_b}}$$

$$\omega_n^2=\frac{K_a}{K_b}, \qquad 2\zeta\omega_n=\frac{1}{K_b}\left(1+\frac{K_a}{a}\right), \qquad \omega_z=a$$

From the above, the natural frequency ω_n, attenuation coefficient ζ, and zero frequency ω_z can be written as follows.

$$\omega_n=\sqrt{\frac{K_a}{K_b}}, \qquad \zeta=\frac{1}{2K_b\sqrt{\frac{K_a}{K_b}}}(1+\frac{K_a}{a}), \qquad \omega_z=a=\frac{\omega_n^2 L}{2\zeta\omega_n L-R}$$

Current PI control gain from this $K_{p_current}$, $K_{i_current}$ will be as follows.

$$K_{p_current}=2\zeta_{CG}\omega_{CG}L-R, \qquad K_{i_current}=K_{p_current}a=\omega_{CG}^2 L$$

ω_{CG}: Current control system intrinsic frequency

ζ_{CG}: Current control system attenuation coefficient

Therefore, we can see that the PI control gain of the current control system can be set by ω_{CG} and ζ_{CG}.

6.3.2 Speed control system design

Model the speed control system from the mechanical properties of the motor. From the equation of motion of the rotating system, the torque equation of the mechanical system can be written as follows.

$$T = J\dot{\omega}_{mech}$$

J : Rotor inertia, ω_{mech}: Mechanical angular velocity

On the other hand, the types of electric torque are as follows when considering only the torque of the magnet.

$$T = P_n \psi_a i_q$$

Using two torque equations, a dynamic system and an electrical system, the mechanical angular velocity can be expressed by the following equation:

$$\omega_{mech} = \frac{P_n \psi_a}{sJ} i_q$$

ω_{mech}: Mechanical angular velocity

Therefore, this is the motor model of the speed control system. In addition, the controller can use PI control and the speed control system can be represented by a feedback control system as shown in Figure 6-5.

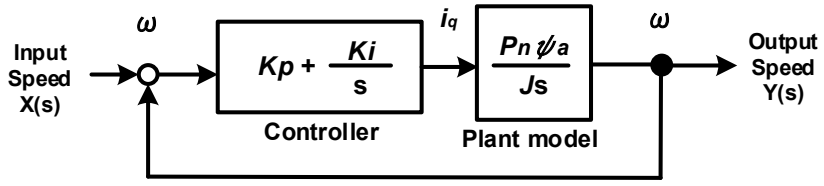


Figure 6-5 Speed control system model

Set the PI control gain of the speed control system, assuming you know the motor parameters P_n, ψ, J. First, find the transfer function of your system.

The closed-loop transfer function of the speed control system is obtained as follows.

$$G(s) = \frac{Y(s)}{X(s)} = \frac{K_b a\left(1 + \frac{s}{a}\right)}{s^2 + K_b s + K_b a}$$

$$K_b = \frac{K_P P_n \psi}{J}, \quad K_i = K_p a$$

In addition, the general formula of the second-order lag system with a zero can be written as follows.

$$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}\left(1 + \frac{s}{\omega_z}\right)$$

Similar to the current control system, if the transfer function of the speed control system is compared with the general formula of the second-order lag system having a zero point, the relational expression as shown below can be obtained.

$$\frac{\omega_n^2(1 + s/\omega_z)}{s^2 + 2\zeta\omega_n s + \omega_n^2} \Leftrightarrow \frac{aK_b\left(1 + \frac{s}{a}\right)}{s^2 + K_b s + aK_b}$$

$$\omega_n^2 = aK_b = \frac{K_p a P_n \psi_a}{J}, \qquad 2\zeta\omega_n = K_b = \frac{K_p P_n \psi_a}{J}, \qquad \omega_z = a$$

From the above, the natural frequency ω_n, attenuation coefficientζ, and zero frequency ω_z can be written as follows.

$$\omega_n = \sqrt{\frac{K_p a P_n \psi_a}{J}}, \qquad \zeta = \frac{1}{2}\sqrt{\frac{K_p P_n \psi_a}{aJ}}, \qquad \omega_z = a = \frac{\omega_n}{2\zeta}$$

From this, the PI control gains K_{p_speed}, K_{i_speed} are as follows.

$$K_{p_speed} = \frac{2\zeta_{SG}\omega_{SG}J}{P_n\psi_a}, \qquad K_{i_speed} = K_{p_speed} * a = \frac{\omega_{SG}^2 J}{P_n\psi_a}$$

ω_{SG}: Speed control system intrinsic frequency

ζ_{SG}: Speed control system damping coefficient

Therefore, we can see that the PI control gain of the speed control system can be designed by ω_{SG} and ζ_{SG}.

6.3.3 Position control system design

Position control system controllers use only proportional terms. In order to respond quickly to excessive input compared to the command value of speed, the responsiveness is improved by combining feedforward to speed. The blocks of the position control system are:

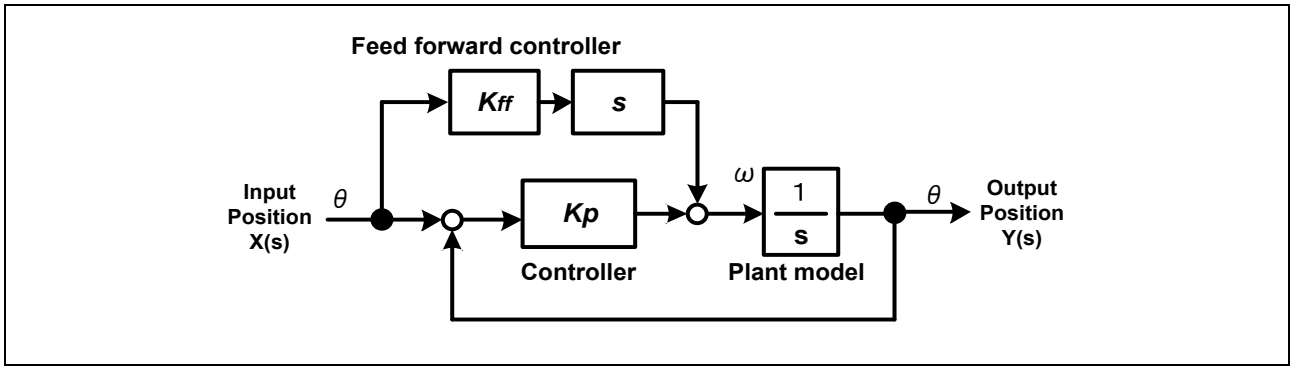


Figure 6-6 Position control system model

The position control system uses only P control, and the $K_{p_position}$ gain design is designed using only the position control system specific frequency ω_{PG}.

$$\omega = K_{p_position}(\theta_{ref} - \theta)$$

$$K_{p_position} = \omega_{PG}$$

In addition, feedforward control to speed is implemented to improve speed responsiveness.

$$\omega_{ff} = \mathrm{K}_{\mathrm{speed_ff}}\,\dot{\theta}_{ref}$$

Therefore, the velocity feedforward can be designed with a fixed value, and the position P gain can be designed with the natural frequency ω_n.

7. Test results

The test results shown in this chapter are reference values measured in the 2.1 Operation check environment.

7.1 Program size

The program size of the sample program is shown in Table 7-1. In the optimization settings of the compiler, optimization level 2 (-optimize = 2) is set, and the optimization method is set to optimization (-size) that emphasizes code size.

Table 7-1 Program size

Memory	Size		
	RX26T RAM64KB Version	RX26T RAM48KB Version	RX14T
ROM	26.8 KB	26.7 KB	24.8 KB
RAM	9.9 KB	9.9 KB	6.1 KB
Maximum value of stack analysis result	380 B	380 B	332 B
Stack size settings	5120 B	5120 B	1536 B

7.2 CPU load factor

The CPU processing time and load factor for each control cycle are shown below.

Table 7-2 Control loops and CPU load factor

CPU Board	Control loop type	Control interval	Processing time	CPU loading rate
RX26T RAM64KB Version	Current control loop	50 us (number of thinning out 0 times)	13.5 us	27.0 %
	Speed/position control loop	500 us	3.5 us	0.7 %
RX26T RAM48KB Version	Current control loop	50 us (number of thinning out 0 times)	15.4 us	30.8 %
	Speed/position control loop	500 us	4.56 us	0.9 %
RX14T	Current control loop	100 µs (number of thinning out 1 times)	34.8 us	34.8 %
	Speed/position control loop	1 ms	10.8 us	1.1 %

7.3 Operating waveform

As a test result, the waveform at the time of control using sample program is shown for reference.

Table 7-3 Measurement conditions

Item	Value	Remarks
Current control system frequency	300 [Hz]	
Current control system attenuation coefficient	1	
Speed control system frequency	12 [Hz]	
Speed control system damping coefficient	1	
Position control system frequency	4 [Hz]	Valid only during position control.
Load	–	Performed with no load

Figure 7-1 shows the result of speed control.

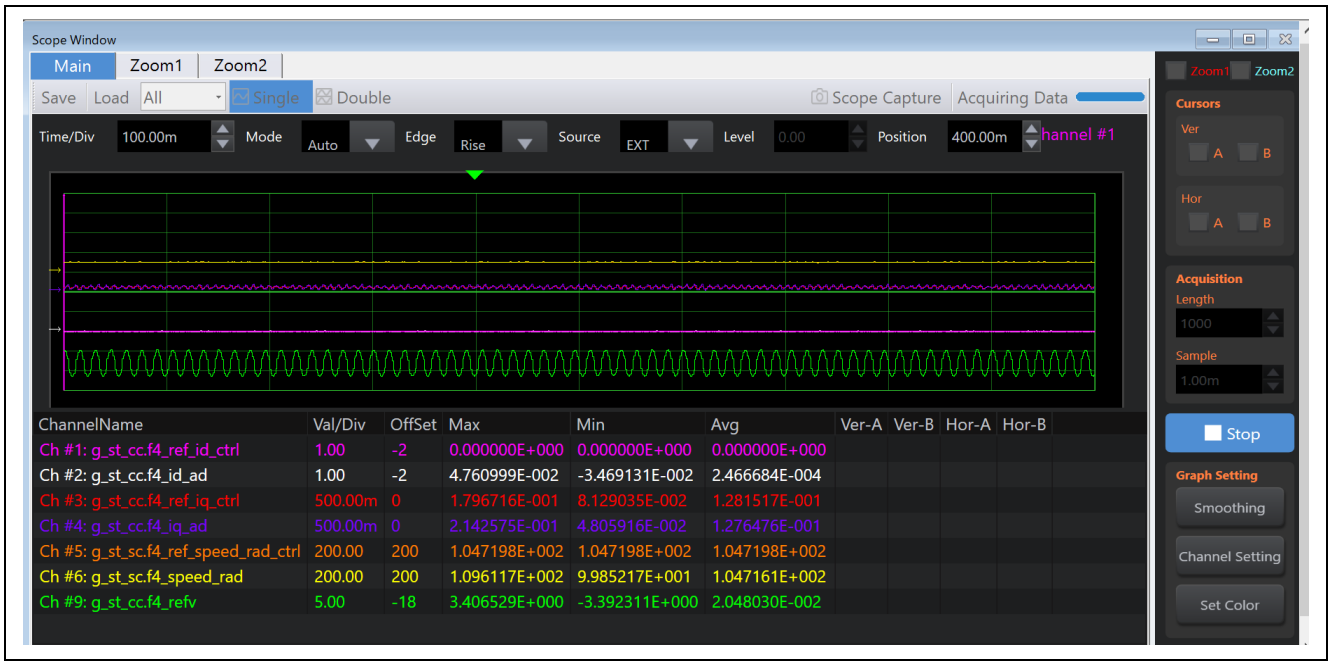


Figure 7-1 Speed control using encoder sensor

Driving conditions:

Rotation speed: Command speed 1000 [rpm]

Waveform information:

Yellow: Detection speed [rad/s] (200rad/s / div.), Orange: Command speed [rad/s] (200rad/s / div.),

Red: q-axis current command value [A] (500mA / div.), Purple: q-axis current value [A] (500mA / div.),

Pink: d-axis current command value [A] (1A / div.), White: d-axis current value [A] (1A / div.),

Yellow green: U-phase voltage[V] (5V / div.),

Horizontal axis: 100ms / div.

Figure 7-2 shows the result of position control.

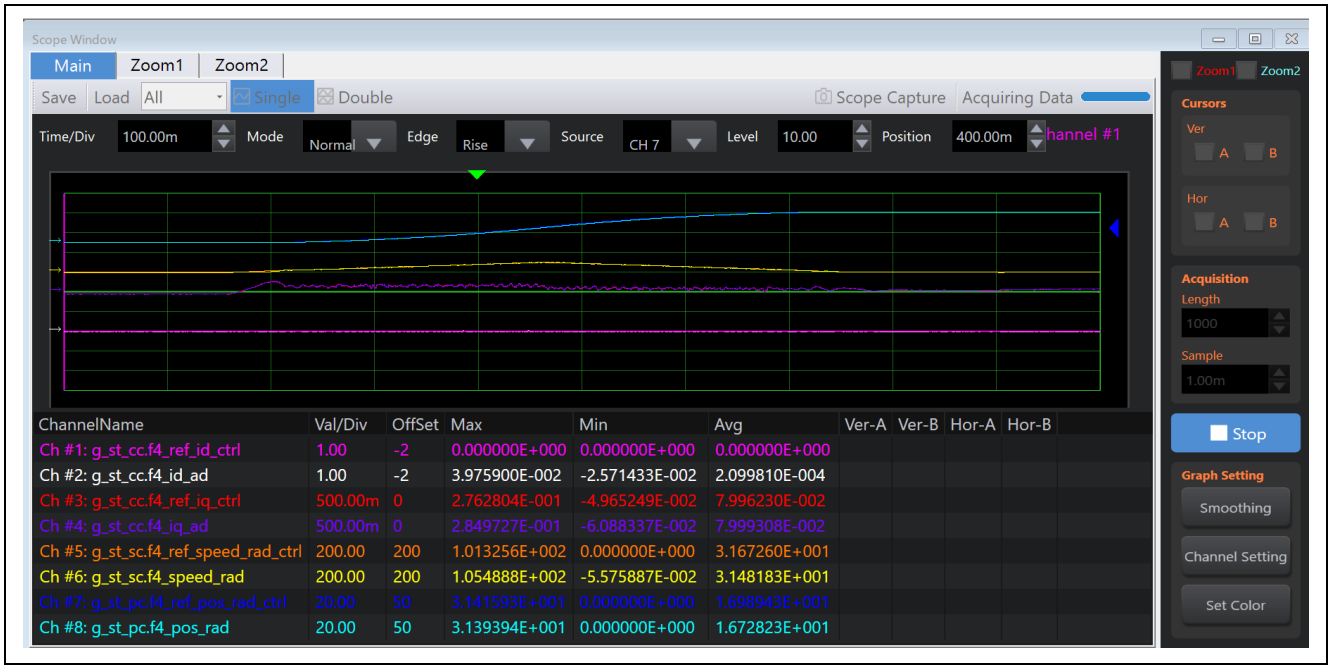


Figure 7-2 Position control using encoder sensor

Driving conditions:

- Position command value: 5 rotations in the CW direction (1800 [degree])
- Maximum position profile speed: 4000 [rpm]
- Acceleration / deceleration time: 300 [ms]

Waveform information:

Yellow: Detection speed [rad/s] (200rad/s / div.), Orange: Command speed [rad/s] (200rad/s / div.),

Red: q-axis current command value [A] (500mA / div.), purple: q-axis current value [A] (500mA / div.),

Pink: d-axis current command value [A] (1A / div.), White: d-axis current value [A] (1A / div.),

Light blue: Angle information calculated from the sensor (machine angle) [rad] (20rad/div.),

Blue: Position command value [rad] (20rad/div.),

Horizontal axis: 100ms / div.

8. Reference materials

- Renesas Motor Workbench User's Manual (R21UZ0004)
- MCK-RX26T User's Manual (R12UZ0111)
- MCK-RX14T User's Manual (R12UZ0192)
- Smart Configurator User's Manual RX API Reference (R20UT4360)
- RX Smart Configurator User Guide: CS + Edition (R20AN0470)
- RX Smart Configurator User Guide: e^2 studio edition (R20AN0451)
- RX26T Group User’s Manual: Hardware (R01UH0979)
- RX14T Group User's Manual: Hardware (R01UH1126)
- MCB-RX26T Type A User's Manual (R12UZ0112)
- MCB-RX26T Type C User's Manual (R12UZ0127)
- MCB-RX14T User's Manual(R12UZ0191)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	May.30.23	—	First edition issued
1.10	Aug.29.23	—	Additional target device(R5F526TACDFM)
1.11	Jun.30.25	P1	Target software version upgrade
		P5	Table 2-1 and Table 2-2 updated
		P7	Addition of Section 2.2.2
		P8	Addition of Section 2.2.3
		P10	Added description of JP11 settings
		P5, P12, P13, P14, P17	Correction of typos
1.20	Jan.30.26	—	Additional target device (R5F514T5AGFM)
1.30	Jun.30.26	P1	Target software version upgrade
		P5	Table 2-1 and Table 2-2 updated
		P5, P8, P9	Table 2-3,Table 2-4,Table2-5,Figure 2-3 and Figure 2-6 added

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

© 2026 Renesas Electronics Corporation. All rights reserved.