Vector Control for Permanent Magnet Synchronous Motor with Magnet Sensor and Inductive Sensor

For the Evaluation System for BLDC Motor

Introduction

This application note is intended to explain the sample program for vector control of a permanent magnet synchronous motor with magnet sensor and inductive sensor, by using functions of a Renesas microcontroller. This program supports digital, analog, and SPI-output magnet sensors and these can be switched by software. In addition, motor control with an inductive sensor is possible by setting the program to analog-output magnet sensor processing. The target software of this application note uses the Smart Configurator tool and the components required for motor control.

The target software of this application note is for reference only, and we do not guarantee the operations. Only use the target software of this application note after conducting thorough evaluation in an appropriate environment.

Target Device

Operations of the target software of this application note are checked by using the following devices.

- MCUs used:
- RX72M (R5F572MNDDBD)
- RX72T (R5F572TKCDFB)
- RX66T (R5F566TEADFP)
- RX24T (R5F524TAADFP)
- RX24U (R5F524UEADFB)
- RX23T (R5F523T5ADFM)
- RX13T (R5F513T5ADFL)

Target Software

The following shows the target software of this application note:

- RXxxx_ESB_SPM_MAG_FOC_CSP_V100 (IDE: CS+ edition)*1 *2
- RXxxx_ESB_SPM_MAG_FOC_E2S_V100 (IDE: e^{2}studio edition)*1 *2
  Evaluation System for BLDC Motor & magnet sensor vector control software for the RXxxx CPU card*1

Notes: 1. The xxx portion is replaced by the name of the MCU to be used.
2. Analog and SPI-output are not supported in RX13T. Analog-output is not supported in RX23T.

Contents

1. Overview

This application note is intended to explain the method of using the sample program for vector control of a permanent magnet synchronous motor with magnet sensor and inductive sensor, by using functions of a Renesas microcontroller. Using the sample program together with a motor control kit (Evaluation System for BLDC Motor) enables motor control. This sample program supports Renesas Motor Workbench, a motor control development support tool, and therefore can be used as a user interface (UI) for checking the MCU internal data and controlling a motor. You can use the sample program for reference purposes when selecting the MCU to be used or developing software by checking how MCU functions are allocated, how control is loaded by interrupts, and other information in the sample program.

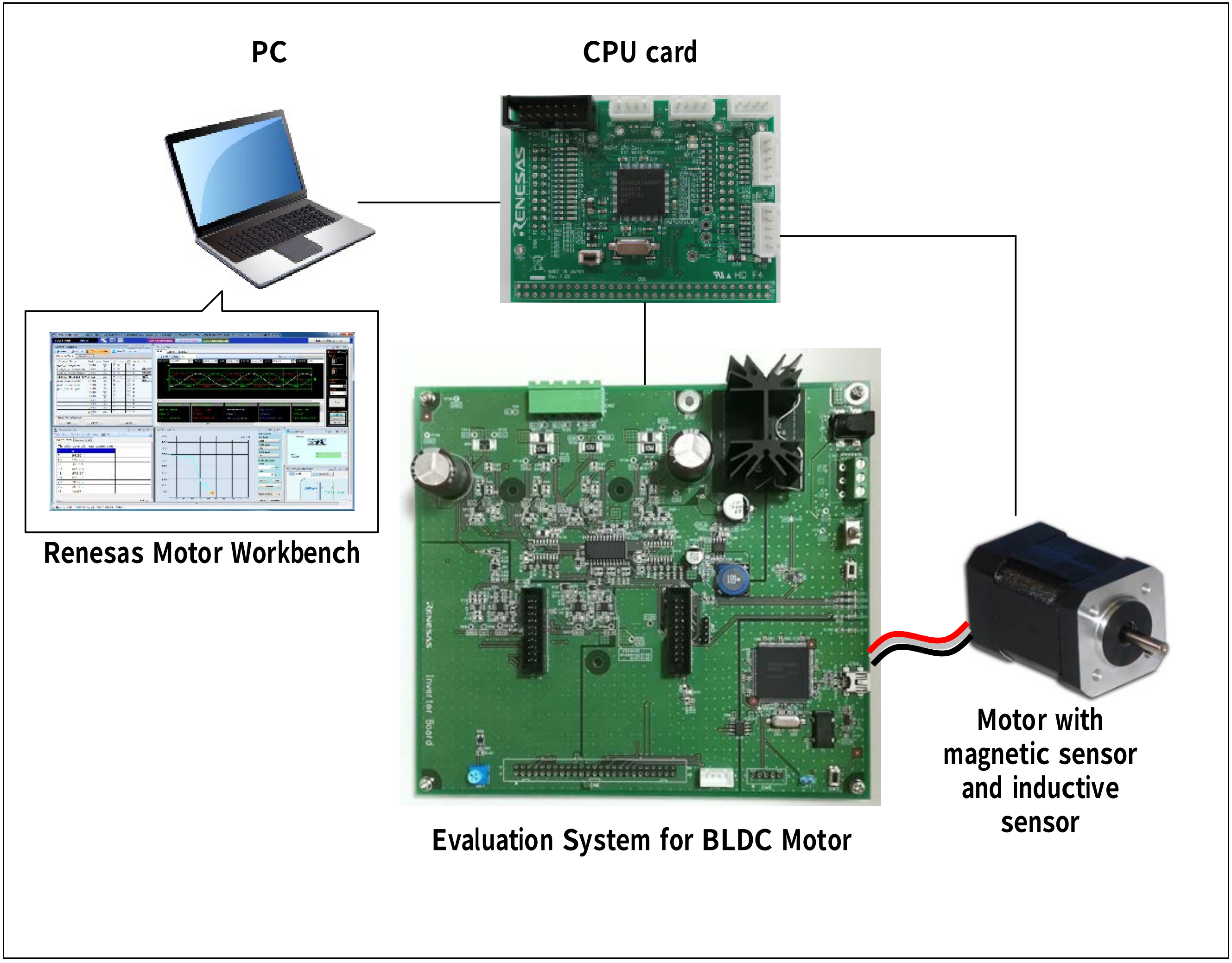


Figure 1-1 Operating Environment of the Sample Program

2. Development Environments

2.1 Test Environments

Table 2-1 and Table 2-2 show the development environments for the software that this application note covers.

Table 2-1 Hardware Development Environment

Category	Product Used
Microcontroller/CPU card product type number	RX72M (R5F572MNDDBD)/RTK0EMXDE0C00000BJ*[1] RX72T (R5F572TKCDFB)/RTK0EMX990C00000BJ RX66T (R5F566TEADFP)/RTK0EMX870C00000BJ RX24T (R5F524TAADFP)/RTK0EM0009C03402BJ RX24U (R5F524UEADFB)/RTK0EMX590C02000BJ RX23T (R5F523T5ADFM)/RTK0EM0003C01202BJ RX13T (R5F513T5ADFL)/RTK0EMXA10C00000BJ
Inverter board	Evaluation System for BLDC Motor (RTK0EM0000B10020BJ) included Inverter board for 48-V, 5-A BLDC
Motor	BLY171D-24V-4000 (manufactured by Anaheim Automation)
Sensor	Magnet sensor: TAD2141 (manufactured by TDK Corporation) Magnet sensor: TAS2143 (manufactured by TDK Corporation) Inductive sensor: IPS2200

Note: 1. Some modifications of the CPU card are required to use SPI-output magnet sensor.

Table 2-2 Software Development Environment

IDE Version	Smart Configurator for RX Microcontrollers	Toolchain Version
CS+: V8.07.00	Version 2.12.0	CC-RX: V3.04.00
e^2studio: 2022-01	Plug-in version of e^2studio	

For the purchase or technical support of this system, contact a Renesas Electronics Corporation sales representative or an authorized Renesas Electronics Corporation product distributor.

2.2 Hardware Specifications

2.2.1 Hardware Configuration Diagram

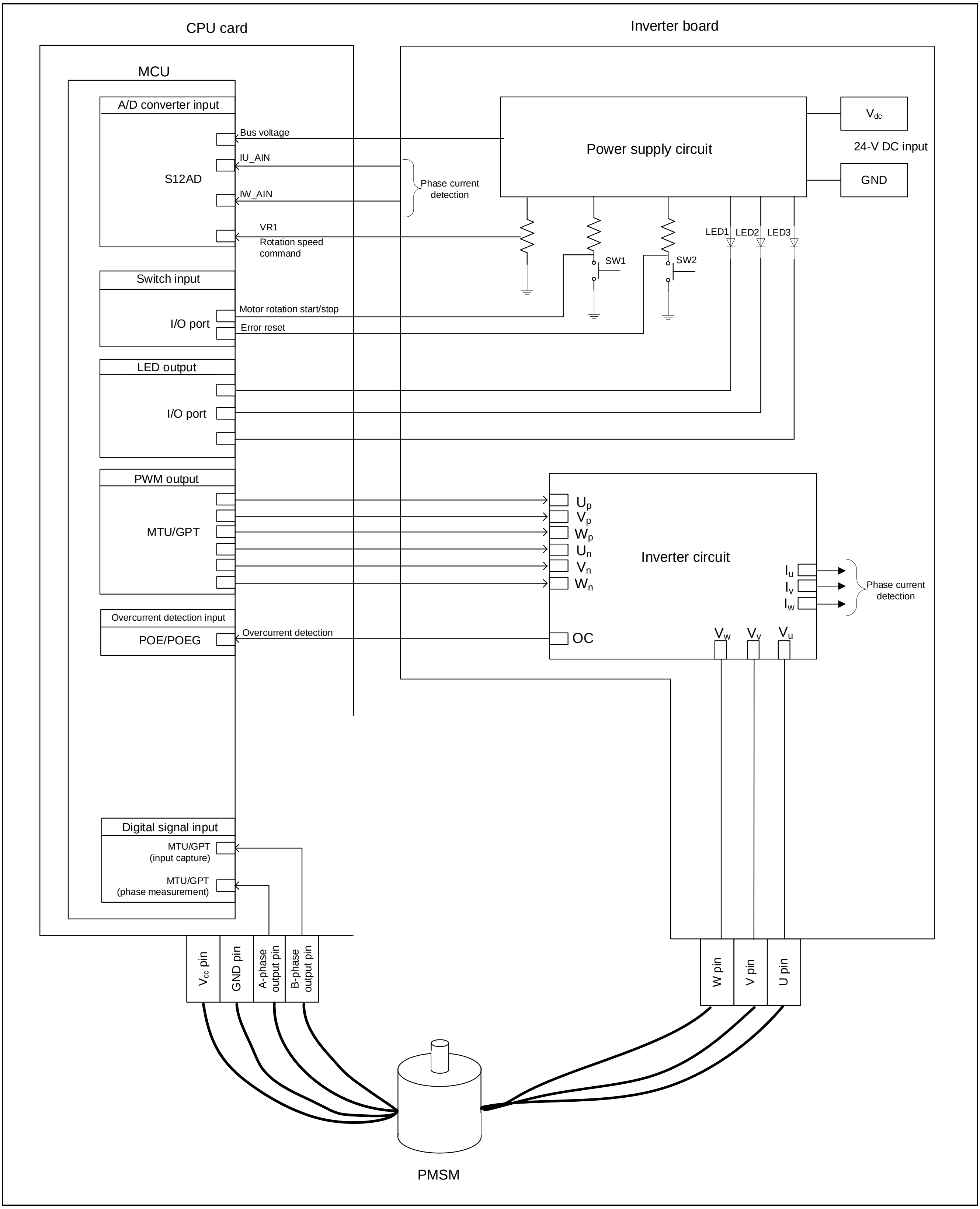


Figure 2-1 Hardware Configuration Diagram (Digital Output Type)

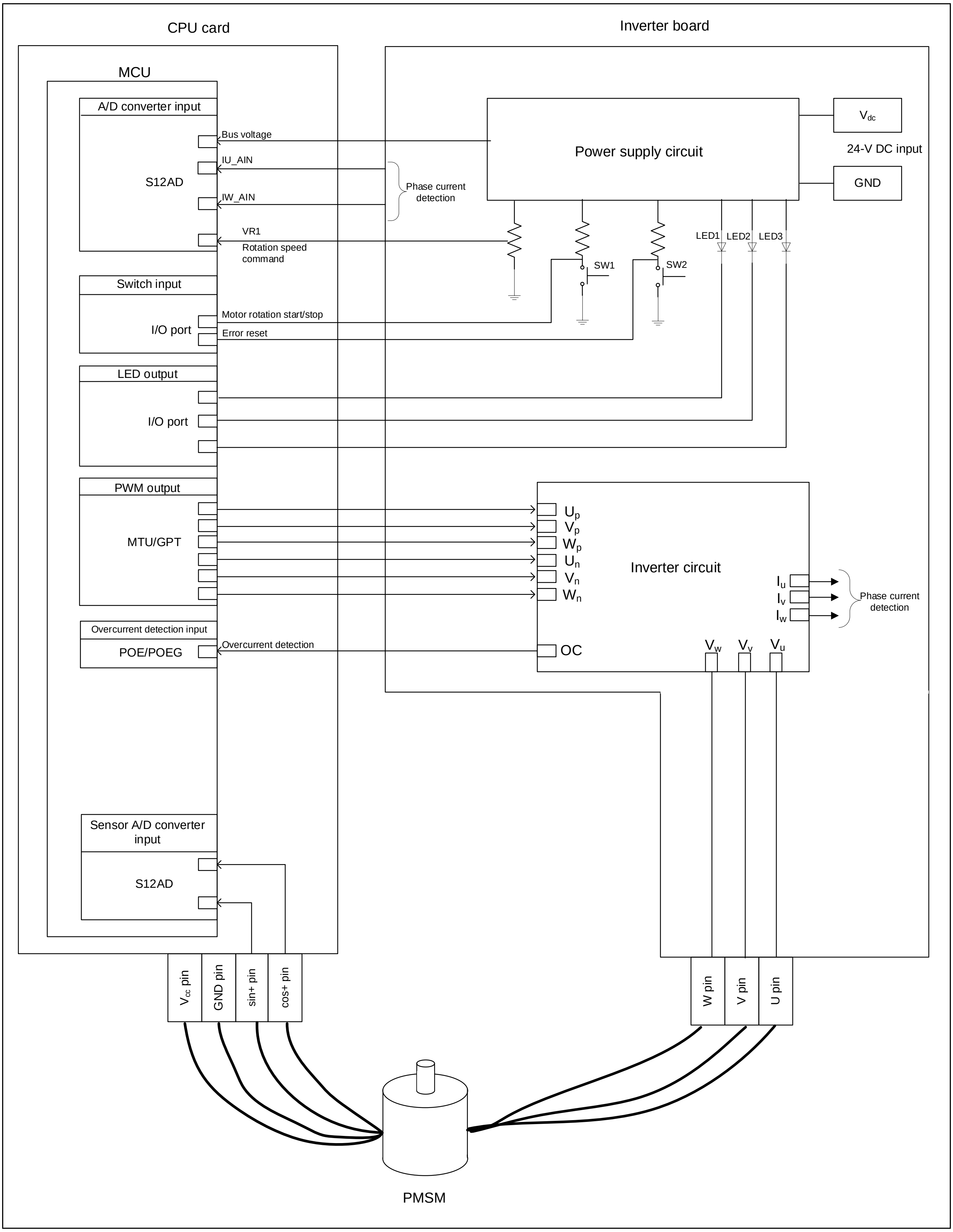


Figure 2-2 Hardware Configuration Diagram (Analog Output Type, Inductive Sensor)

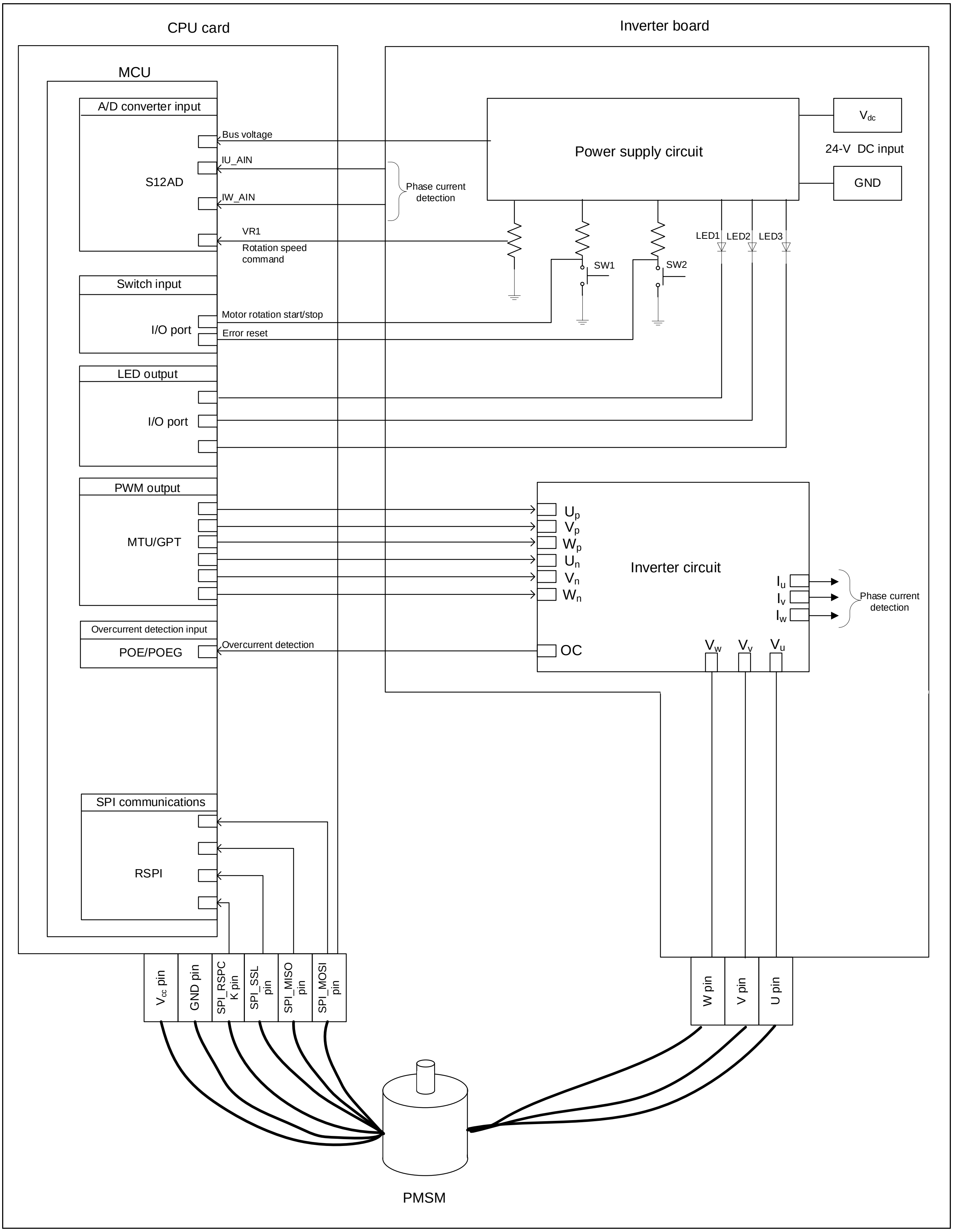


Figure 2-3 Hardware Configuration Diagram (SPI Output Type)

2.2.2 Board User Interface

Table 2-3 lists the components of the user interface of the board for this system.

Table 2-3 Board User Interface

Item	Interface Component	Function
Rotation position/speed	Volume (VR1)	Inputs the rotation position/speed command value (analog value).
START/STOP	Toggle switch (SW1)	Instructs start or stop of motor rotation.
ERROR RESET	Push-button switch (SW2)	Instructs recovery from an error state
LED1	Orange LED	On: The motor is rotating. Off: The motor is stopped.
LED2	Orange LED	On: An error was detected. Off: The system is operating normally.
LED3	Orange LED	On: Positioning is complete. Off: Positioning is not complete.
RESET	Push-button switch (RESET1)	System reset

2.2.3 Peripheral Functions

Table 2-4 shows allocation of input/output functions to peripheral functions that are used in this system. Because pin allocation differs depending on the CPU card that is used, allocation of MCU peripheral functions also differs. In the sample program, the Smart Configurator is used to configure the peripheral functions. For details, see 5.14.

Table 2-4 Input/Output Functions and Peripheral Functions

Function	Peripheral Function
Measurement of the inverter bus voltage	S12AD
Function for inputting the rotation position/speed command value (analog value)	S12AD
START/STOP toggle switch	I/O port (input)
Controlling whether to turn on LED1	I/O port (output)
Controlling whether to turn on LED2	I/O port (output)
Controlling whether to turn on LED3	I/O port (output)
Measurement of the U-phase current	S12AD
Measurement of the W-phase current	S12AD
PWM output (U_p)/active low	MTU/GPT *1
PWM output (V_p)/active low	MTU/GPT *1
PWM output (W_p)/active low	MTU/GPT *1
PWM output (U_n)/active high	MTU/GPT *1
PWM output (V_n)/active high	MTU/GPT *1
PWM output (W_n)/active high	MTU/GPT *1
Sensor digital output detection	MTU
Sensor SPI communications	RSPI
Sensor analog output detection	S12AD
PWM emergency stop input when an overcurrent is detected	POE/POEG *2

Notes: 1. The peripheral function that performs PWM output differs depending on the CPU card.
2. The peripheral function to be used differs depending on the CPU card to interlock the overcurrent signal input and the PWM output stop request.

3. Quick Start Guide

This chapter provides a quick start guide for you to drive a motor by using Evaluation System for BLDC Motor and the sample program. For details about the board configuration and connection procedures of Evaluation System for BLDC Motor, see the Evaluation System for BLDC Motor User's Manual (R12UZ0062). For details about how to use the Renesas Motor Workbench (RMW), see the Renesas Motor Workbench User's Manual (R21UZ0004).

3.1 Downloading and Writing the Sample Program

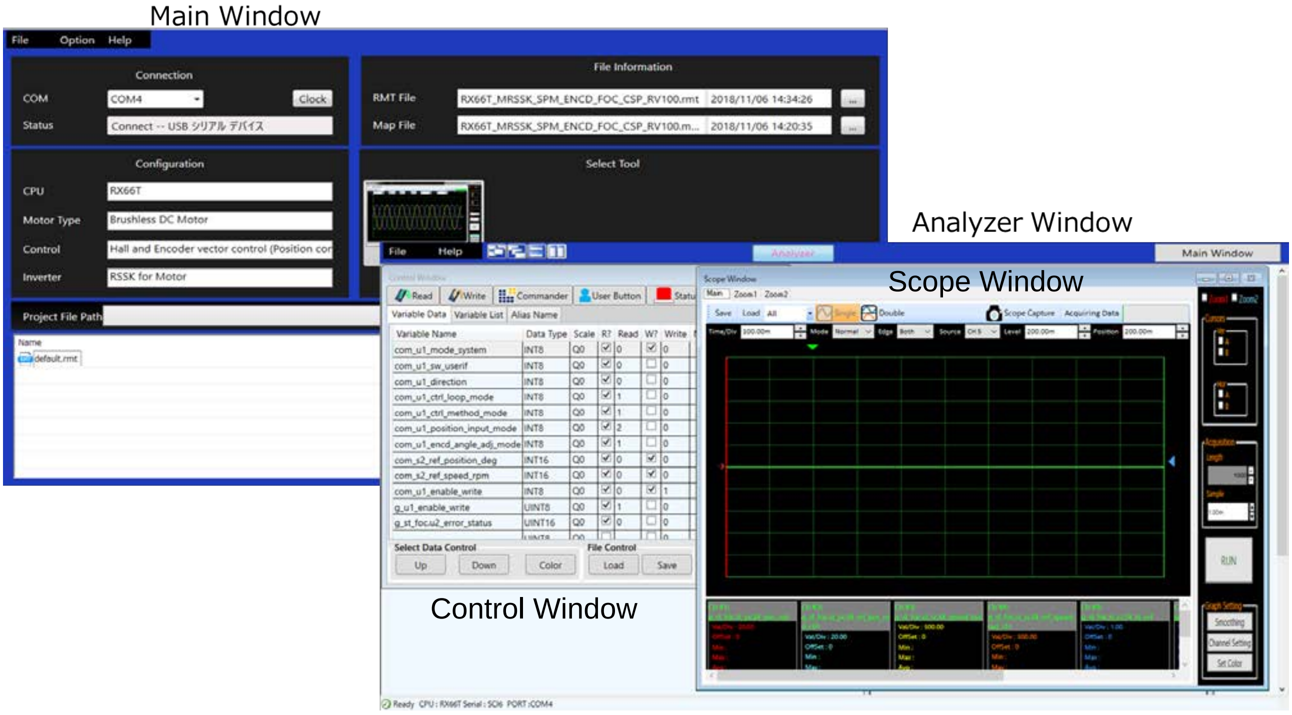
Download the sample program from our website and open it in an integrated development environment (IDE). Set SENSOR_CFG_MR_CONTROL (defined in r_motor_module_cfg.h in the cfg folder under the motor_module folder) according to the magnet sensor to be used, and then recompile the program.

- For analog output magnet sensor and inductive sensor: SENSOR_CFG_MR_CONTROL_ANALOG
- For digital output magnet sensor: SENSOR_CFG_MR_CONTROL_DIGITAL
- For SPI output magnet sensor: SENSOR_CFG_MR_CONTROL_SPI

After the sample program is compiled, use the IDE or Renesas Flash Programmer to write it to the MCU on the CPU card. For details about how to write programs, see the documentation for the IDE you use or Renesas Flash Programmer. If the CPU card does not include E2 On-Board, a dedicated emulator is necessary for writing the programs.

3.2 Analyzer Startup and the RMT File

Use Renesas Motor Workbench, a motor control development support tool, as a user interface (for issuing the rotation start/stop command, rotation speed command, and other commands). The Renesas Motor Workbench (RMW) can be downloaded from our website.


Main Window

Analyzer Window

Scope Window

Control Window

COM: COM4
Status: Connect -- USB シリアル デバイス
RMT File: RX66T_MRSSK_SPM_ENCD_FOC_CSP_RV100.rmt 2018/11/06 14:34:26
Map File: RX66T_MRSSK_SPM_ENCD_FOC_CSP_RV100.m... 2018/11/06 14:20:35
CPU: RX66T
Motor Type: Brushless DC Motor
Control: Hall and Encoder vector control (Position cor
Inverter: RSSK for Motor
Name: default.rmt

Variable Name	Data Type	Scale	R?	Read	W?	Write
com_u1_mode_system	INT8	Q0	☑	0	☑	0
com_u1_sw_userif	INT8	Q0	☑	0	☐	0
com_u1_direction	INT8	Q0	☑	0	☐	0
com_u1_ctrl_loop_mode	INT8	Q0	☑	1	☐	0
com_u1_ctrl_method_mode	INT8	Q0	☑	1	☐	0
com_u1_position_input_mode	INT8	Q0	☑	2	☐	0
com_u1_encd_angle_adj_mode	INT8	Q0	☑	1	☐	0
com_s2_ref_position_deg	INT16	Q0	☑	0	☑	0
com_s2_ref_speed_rpm	INT16	Q0	☑	0	☑	0
com_u1_enable_write	INT8	Q0	☑	0	☑	1
g_u1_enable_write	UINT8	Q0	☑	1	☐	0
g_st_foc.u2_error_status	UINT16	Q0	☑	0	☐	0

Figure 3-1 Windows of Renesas Motor Workbench

How to use Renesas Motor Workbench (motor control development support tool)

- Click the Renesas Motor Workbench icon to start the tool.
- On the menu bar of the Main Window, select [File] > [Open RMT File(O)].
  The RMT file in the "rmw" folder in the project folder is loaded.
- In the [Connection] area, from the [COM] drop-down list, select the COM of the connected kit.
- In the [Select Tool] area, click the [Analyzer] button to open the Analyzer Window.
- Start driving the motor as described in Using the RMW UI. (For details, see 3.4.)

What is the RMT file?

- The RMT file is a file that holds the environmental information that was manipulated or configured by using the RMW.
- If the environmental information has been saved in the RMT file, the environment can be restored with the saved information by calling the RMT file.
- If the address information of a program is changed, load the map file that was generated during program building, and then save the RMT file again.

3.3 List of Variables for Analyzer Functions

Table 3-1 lists the data input variables that are used when the RMW UI is used. The values input to these variables are applied to the corresponding variables in the motor module and then used for controlling the motor if the value written to the com_u1_enable_write variable is the same as the value of the g_u1_enable_write variable. Note, however, that the variables indicated by an asterisk (*) do not depend on the value of the com_u1_enable_write variable.

Table 3-1 List of Main Input Variables for Analyzer Functions

Analyzer Function Input Variable Name	Type	Description
com_u1_sw_userif (*)	uint8_t	Switching of the user interface 0: Uses the RMW UI. (Default) 1: Uses the board UI.
com_u1_system_mode (*)	uint8_t	Managing the state 0: Stop mode 1: Run mode 3: Reset
com_u1_ctrl_loop_mode	uint8_t	Switching of the control loop 0: Position control 1: Speed control (default)
com_s2_ref_position_deg (*)	int16_t	Position command value (mechanical angle) [degrees]
com_s2_ref_speed_rpm (*)	int16_t	Speed command value (mechanical angle) [rpm]
com_u1_enable_write	uint8_t	Whether to enable rewriting of variables for user entry The input data is applied if the values of this and g_u1_enable_write variables are the same.

Table 3-2 lists main structure variables that are often monitored when the driving under position or speed control is evaluated using a magnet sensor. Use this table for reference when the waveform is to be displayed or the values of variables are to be loaded with an analyzer function. For details about the variables that are not listed in this table, see 5.1.5.

Table 3-2 List of Main Variables for Position or Speed Control

Main Position/Speed Control Variable Name	Type	Description
g_st_mr_vector.u2_error_status	uint16_t	Error state
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 (mechanical angle) [rad/s]
g_st_sc.f4_speed_rad	float	Speed detection value (mechanical angle) [rad/s]
g_st_pc.f4_ref_pos_rad_ctrl	float	Position command value (mechanical angle) [rad]
g_st_pc.f4_pos_rad	float	Position detection value (mechanical angle) [rad]

3.4 Using the RMW UI

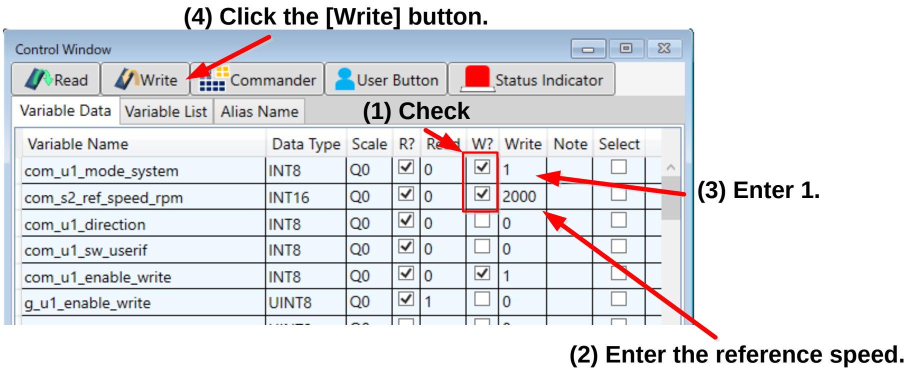
3.4.1 Analyzer Operation Example

The following shows an example of using the analyzer function to perform operations on the motor. The operations are performed from the Control Window. For details about the Control Window, see the Renesas Motor Workbench User's Manual.

In the initial state, the control loop is set for speed control. Perform operations by referring to the procedures shown below.

(a) Starting rotation of the motor

(1) Confirm that the check boxes in the [W?] column are selected on the "com_u1_mode_system" and "com_s2_ref_speed_rpm" rows.
(2) On the "com_s2_ref_speed_rpm" row, in the [Write] column, enter the command rotation speed.
(3) On the "com_u1_mode_system" row, in the [Write] column, enter 1.
(4) Click the [Write] button.



Variable Name	Data Type	Scale	R?	Read	W?	Write	Note	Select
com_u1_mode_system	INT8	Q0	☑	0	☑	1		☐
com_s2_ref_speed_rpm	INT16	Q0	☑	0	☑	2000		☐
com_u1_direction	INT8	Q0	☑	0	☐	0		☐
com_u1_sw_userif	INT8	Q0	☑	0	☐	0		☐
com_u1_enable_write	INT8	Q0	☑	0	☑	1		☐
g_u1_enable_write	UINT8	Q0	☑	1	☐	0		☐

(4) Click the [Write] button.

(1) Check

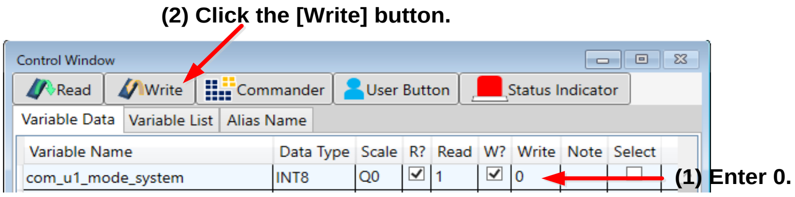
(3) Enter 1.

(2) Enter the reference speed.

Figure 3-2 Procedure for Starting Rotation of the Motor

(b) Stopping the motor

(1) On the "com_u1_mode_system" row, in the [Write] column, enter 0.
(2) Click the [Write] button.



Variable Name	Data Type	Scale	R?	Read	W?	Write	Note	Select
com_u1_mode_system	INT8	Q0	☑	1	☑	0		☐

(2) Click the [Write] button.

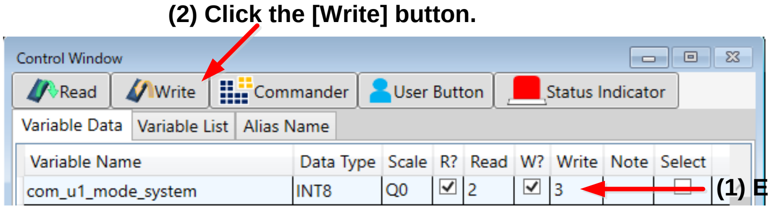
(1) Enter 0.

Figure 3-3 Procedure for Stopping the Motor

(c) What to do in case of motor stoppage (due to an error)

(1) On the "com_u1_mode_system" row, in the [Write] column, enter 3.

(2) Click the [Write] button.



Variable Name	Data Type	Scale	R?	Read	W?	Write	Note	Select
com_u1_mode_system	INT8	Q0	☑	2	☑	3		☐

(2) Click the [Write] button.

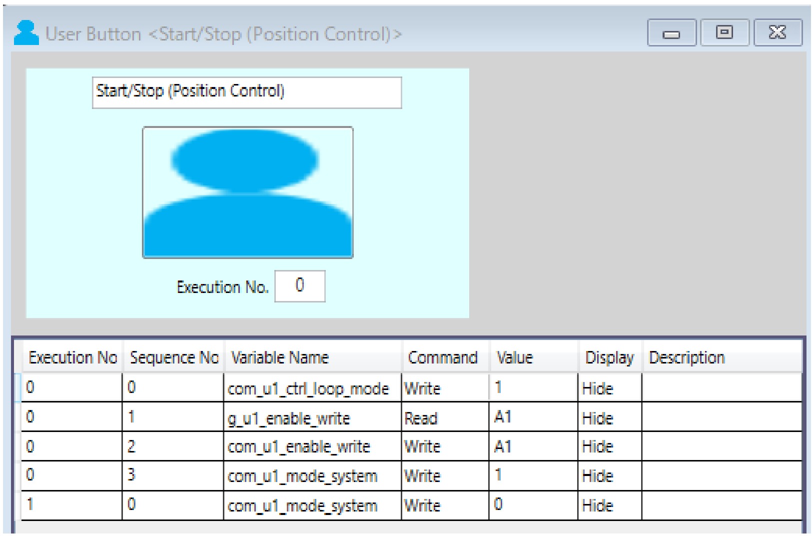
(1) Enter 3.

Figure 3-4 Procedure for Handling an Error

3.4.2 Operation Example of the User Button Function

The following shows an example of using the user button function to perform operations on the motor. The user button settings used in this example are already included in the RMT file in the sample program.

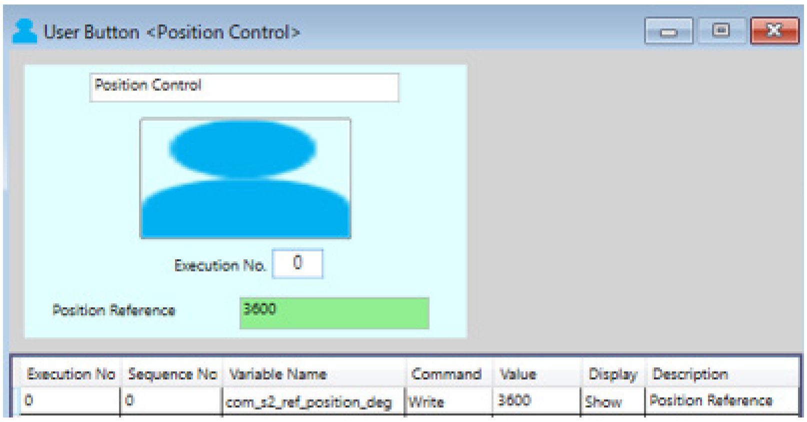
- Starting or stopping motor rotation by position control
  By specifying settings as shown in Figure 3-5, each click of the button switches between starting and stopping.



Execution No	Sequence No	Variable Name	Command	Value	Display	Description
0	0	com_u1_ctrl_loop_mode	Write	1	Hide	
0	1	g_u1_enable_write	Read	A1	Hide	
0	2	com_u1_enable_write	Write	A1	Hide	
0	3	com_u1_mode_system	Write	1	Hide	
1	0	com_u1_mode_system	Write	0	Hide	

Figure 3-5 Starting or Stopping Motor Rotation

- Changing the position command
  By specifying settings as shown in Figure 3-6, the position can be changed by entering the desired information and then clicking the button.

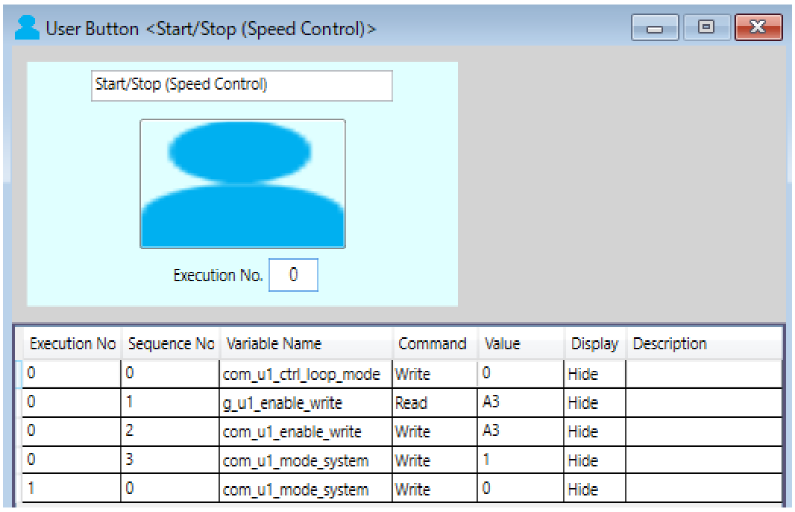


Position Reference 3600

Execution No	Sequence No	Variable Name	Command	Value	Display	Description
0	0	com_s2_ref_position_deg	Write	3600	Show	Position Reference

Figure 3-6 Changing the Position Command

- Starting or stopping motor rotation by speed control
  By specifying settings as shown in Figure 3-7, each click of the button switches between starting and stopping.


User Button <Start/Stop (Speed Control)>

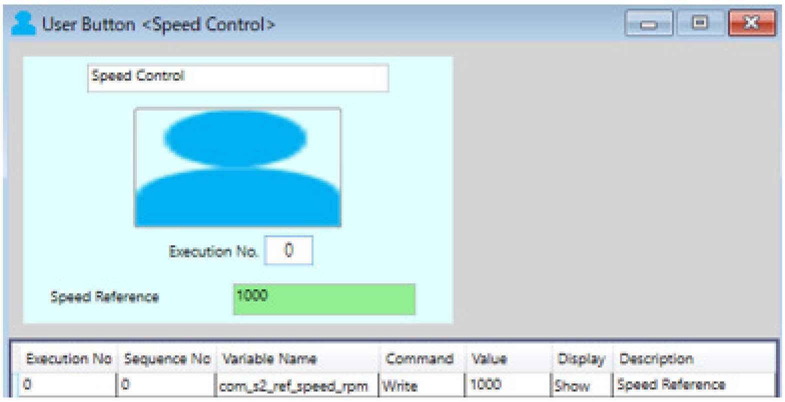
Start/Stop (Speed Control)

Execution No. 0

Execution No	Sequence No	Variable Name	Command	Value	Display	Description
0	0	com_u1_ctrl_loop_mode	Write	0	Hide	
0	1	g_u1_enable_write	Read	A3	Hide	
0	2	com_u1_enable_write	Write	A3	Hide	
0	3	com_u1_mode_system	Write	1	Hide	
1	0	com_u1_mode_system	Write	0	Hide	

Figure 3-7 Starting or Stopping Motor Rotation

- Changing the speed command
  By specifying settings as shown in Figure 3-8, the speed command can be changed by entering the desired information and then clicking the button.


User Button <Speed Control>

Speed Control

Execution No. 0

Speed Reference 1000

Execution No	Sequence No	Variable Name	Command	Value	Display	Description
0	0	com_s2_ref_speed_rpm	Write	1000	Show	Speed Reference

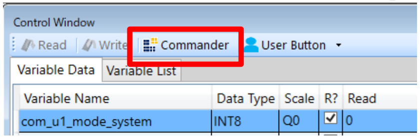
Figure 3-8 Changing the Speed Command

3.4.3 Operation Example of the Commander Function

Using the commander function to perform position control

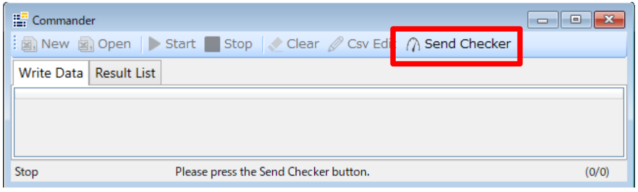
Starting the commander:

(1) In the Control Window, click the [Commander] button.



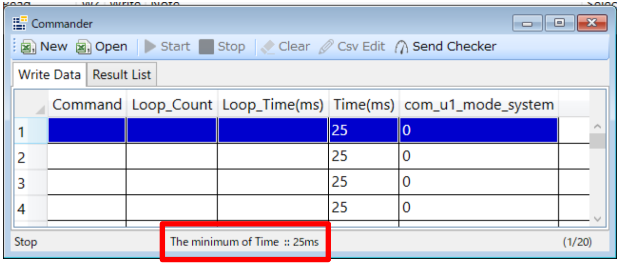
Variable Name	Data Type	Scale	R?	Read
com_u1_mode_system	INT8	Q0	☑	0

(2) When the Commander window appears, click the [Send Checker] button, and then check the data transmission speed.



Stop Please press the Send Checker button. (0/0)

The minimum transmission time is displayed.

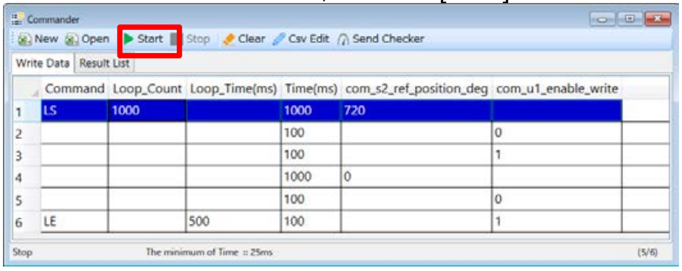


	Command	Loop_Count	Loop_Time(ms)	Time(ms)	com_u1_mode_system
1				25	0
2				25	0
3				25	0
4				25	0

Stop The minimum of Time :: 25ms (1/20)

(3) Click the [Open] button to load "Position_test.csv". Place the system in the position control mode, write 1 to com_u1_mode_system, and then click the [Write] button so that the system enters the run mode.
The motor starts positioning control.

(4) In the Commander window, click the [Start] button to start a sequence.



	Command	Loop_Count	Loop_Time(ms)	Time(ms)	com_s2_ref_position_deg	com_u1_enable_write
1	LS	1000		1000	720	
2				100		0
3				100		1
4				1000	0	
5				100		0
6	LE		500	100		1

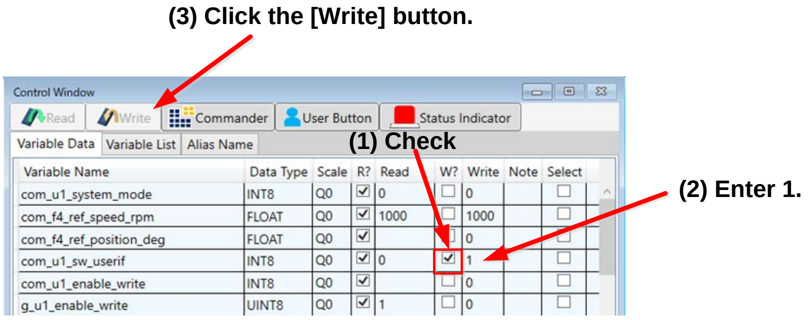
Stop The minimum of Time :: 25ms (5/6)

3.5 Using the Board UI

3.5.1 Switching the User Interface

In the sample program, the RMW UI has been set as the user interface. To change the user interface to the board UI, perform the following procedure.

On the "com_u1_sw_userif" row, confirm that the check box in the [W?] column is selected, and then enter 1 in the [Write] column. Click the [Write] button.



(3) Click the [Write] button.

(1) Check

(2) Enter 1.

Variable Name	Data Type	Scale	R?	Read	W?	Write	Note	Select
com_u1_system_mode	INT8	Q0	☑	0	☐	0		☐
com_f4_ref_speed_rpm	FLOAT	Q0	☑	1000	☐	1000		☐
com_f4_ref_position_deg	FLOAT	Q0	☑		☐	0		☐
com_u1_sw_userif	INT8	Q0	☑	0	☑	1		☐
com_u1_enable_write	INT8	Q0	☑		☐	0		☐
g_u1_enable_write	UINT8	Q0	☑	1	☐	0		☐

Figure 3-9 Procedure for Switching the UI

3.5.2 Starting or Stopping the Motor

If the board UI is used, start and stop of motor rotation are controlled by the input from SW1 on the inverter board (via the board UI). A general-purpose port is assigned to SW1. The system determines whether to start or stop motor rotation by reading a pin in the main loop. If the pin is driven low, the system judges that the START switch is pressed. If the pin is driven high, the system judges that the motor is to be stopped.

3.5.3 Motor Rotation Position/Speed Command Value

The motor rotation position/speed command value is determined by performing A/D conversion for the output value (analog value) of VR1 on the inverter board. The VR1 value after A/D conversion is used as the rotation position/speed command value as shown in the following table.

Table 3-3 Conversion Ratio of the Command Value

Item	**Conversion Ratio (Command Value : Value after A/D Conversion)**	
Rotation position command value	CW	0 to 180 [degrees]: 07FFH to 0000H
	CCW	0 to -180 [degrees]: 0800H to 0FFFH
Rotation speed command value	CW	0 to 4000 [rpm]: 07FFH to 0000H
	CCW	0 to -4000 [rpm]: 0800H to 0FFFH

4. Software

4.1 Software Specifications

The following shows the basic software specifications of this system.

Table 4-1 Basic Specifications of Magnet Sensor Vector Control Software

Item	Description	
Control method	Vector control	
Starting/stopping motor control	Determined by the level of SW1 (low: start control; high: stop) or input from the RMW	
Rotor magnetic pole position detection	Magnet sensor (analog, digital, and SPI output)	
Input voltage	24-V DC	
Carrier frequency (PWM)	20 [kHz], Carrier cycle: 50 [μs] (RX13T: 16 kHz, Carrier cycle: 62.5 [μs])	
Dead time	2 [μs]	
Control cycle (current)	RX72M/RX72T/RX66T: 50 [μs] RX23T/RX24T/RX24U: 100 [μs] (twice the carrier cycle) RX13T: 125 [μs] (twice the carrier cycle)	
Control cycle (speed/position)	RX72M/RX72T/RX66T: 500 [μs] RX13T/RX23T/RX24T/RX24U: 1 [ms]	
Position command value management	Board UI	Creation of position command values: Directly input by VR1 Input range: -180° to 180°
	RMW UI	Creation of position command values: Position profiling by using the speed trapezoidal wave method Input range: -32768° to 32767° CW/CCW speed limitation: -4000 to 4000 [rpm]
Speed command value management	CW: 0 to 4000 [rpm] CCW: 0 to -4000 [rpm]	
Position resolution	Digital: 0.02° (16384 ppr) Analog: (Sensor output [V]/5 [V]) x 4096 [ppr] SPI: 0.08° (12-bit output)	
Dead band of the position*1	Digital: Digital pulse ± 1 count (±0.09°)	
Natural frequency for each control system	Current control system: 300 Hz, speed control system: 30 Hz, position control system: 10 Hz	
Compiler optimization settings	Optimization level	2 (-optimize = 2) (default)
	Optimization method	Optimization focusing on the code size (-size) (default)
Protection stop processing	The motor control signal output (six outputs) will go to the inactive level when any of the following conditions is met: 1. The currents of all phases exceed 3.82 [A] (checked at intervals of 50 [μs]). 2. The inverter bus voltage exceeds 28 [V] (checked at intervals of 50 [μs]). 3. The inverter bus voltage is less than 14 [V] (checked at intervals of 50 [μs]). 4. The rotation speed exceeds 4500 [rpm] (checked at intervals of 50 [μs]). When the overcurrent detection signal (POE/POEG) from an external circuit or an output short-circuit is detected, the PWM output pin is driven to high impedance.	

Note: 1. A dead band is provided to prevent hunting at the time of positioning.

4.2 Software Configuration

The sample program consists of the application layer, motor module, and Smart Configurator. The motor module controls the motor as requested from the application layer controlled by the user. The output from the motor module is transferred by the Smart Configurator to the hardware layer.

4.2.1 Overall Configuration

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

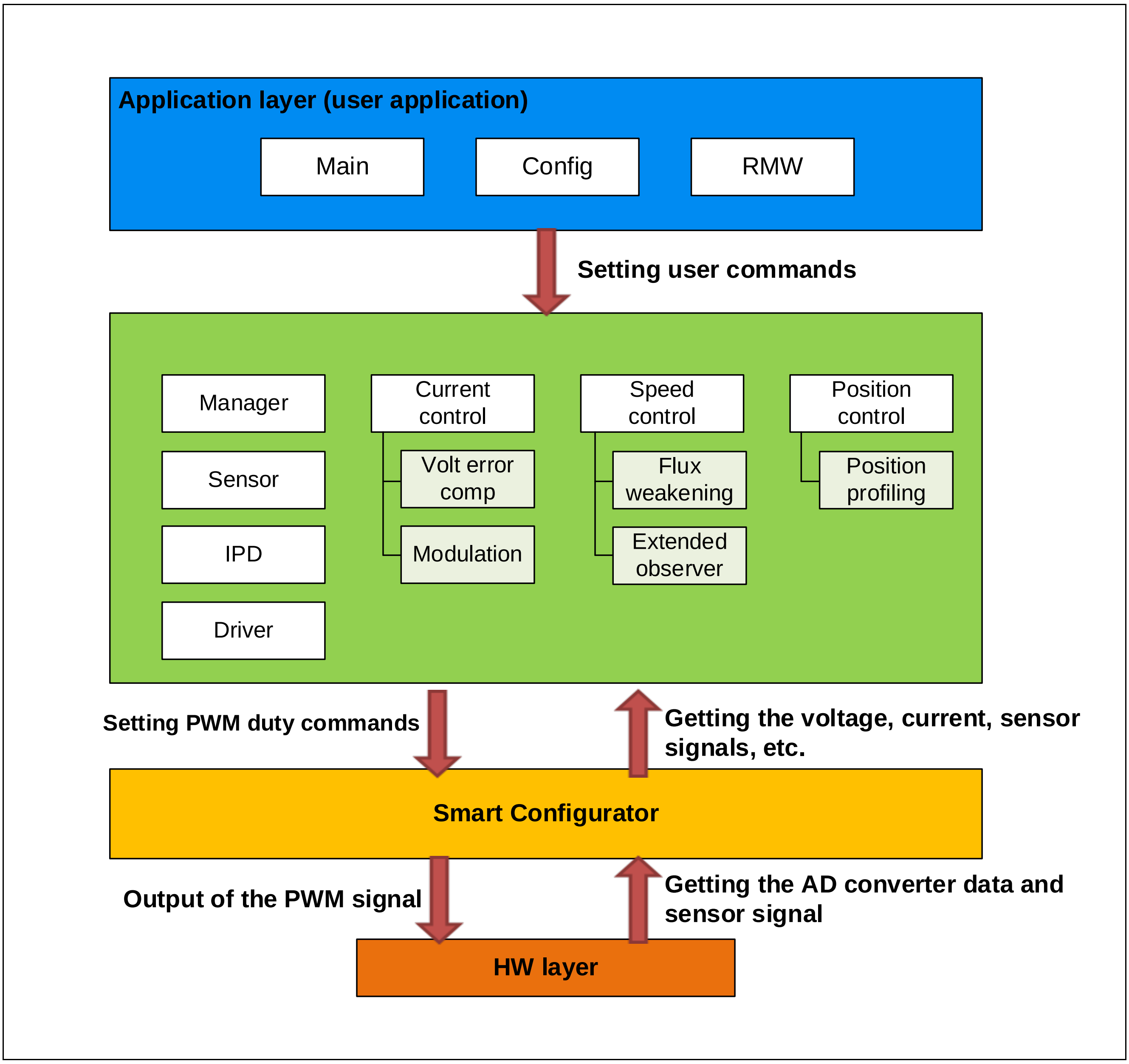


Figure 4-1 Overall Configuration of the Motor Control Software

4.2.2 Configuration of the Motor Module

Figure 4-2 shows the configuration of the motor module. Table 4-2 provides an outline of each module. The manager module works as an interface between other modules and performs data acquisition and setting for the appropriate modules.

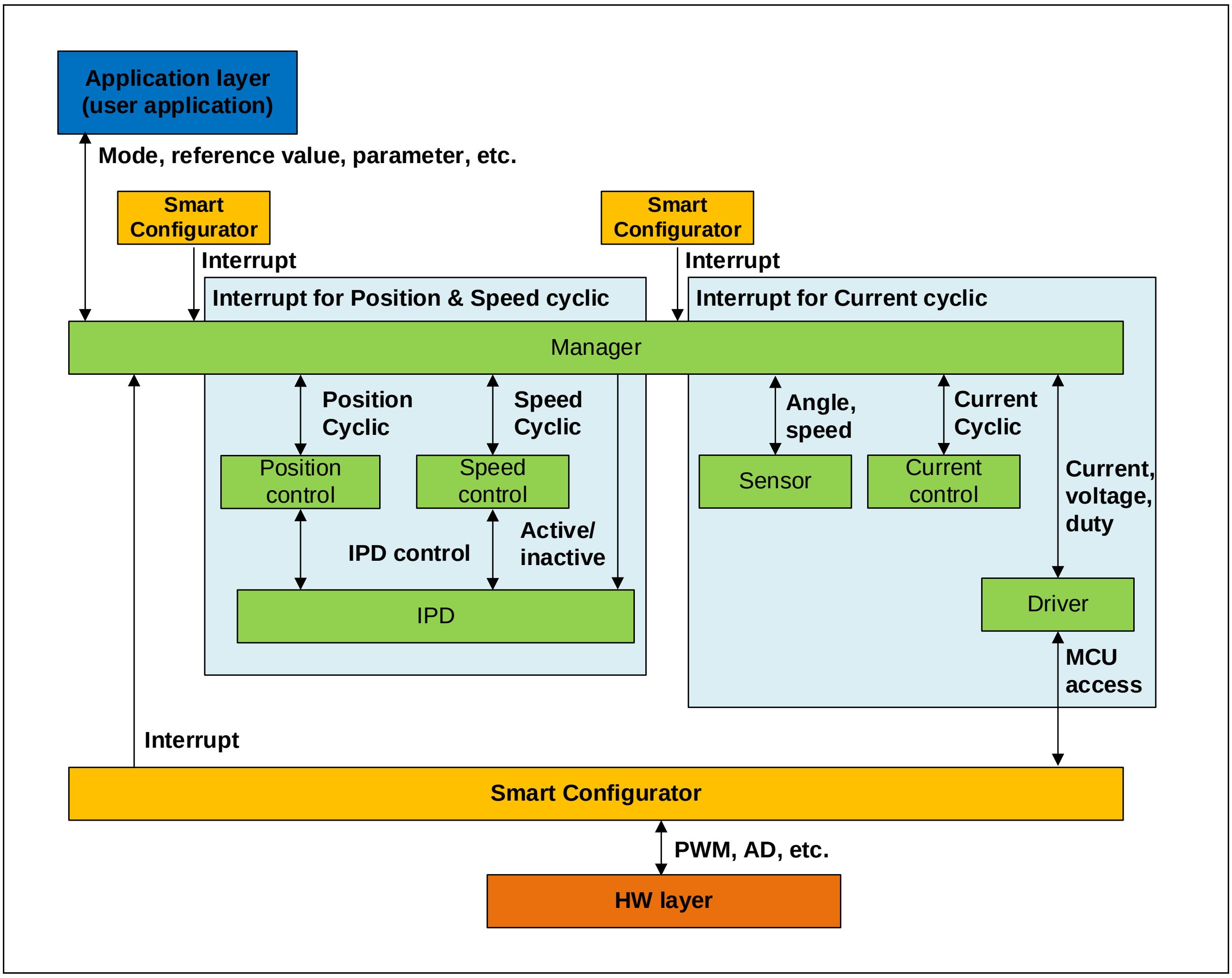


Figure 4-2 Configuration of the Motor Module

Table 4-2 Outline of the Modules

Module	Description	Section
Application layer	Main processing and user area	5.1
Manager module	Management of the overall sample program and interface with each module	5.2
Current control module	Module related to current control	5.3
Speed control module	Module related to speed control	5.6
Position control module	Module related to position control	5.9
IPD control module	Module related to IPD control	5.11
Sensor module	Module for acquiring position and speed information from sensor signals	5.12
Driver module	Module related to connection with the Smart Configurator	5.13
Smart Configurator layer	Module related to connection with the hardware layer	5.14

4.3 File and Folder Configuration

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

Table 4-3 File and Folder Configuration

Folder	Subfolder	File	Remarks
app	main	r_app_main.c/h	User main function
	rmw	r_app_rmw.c/h	Definition of functions related to the RMW analyzer UI
		r_app_rmw_interrupt.c	Definition of RMW interrupt functions
		ICS2_RX"xxx".lib/h ICS_RX"xxx".obj/h	Library for RMW communications
	board_ui	r_app_board_ui.c/h	Definition of functions related to the board UI
		r_app_board_ui_ctrl.h	Definition of MCU-dependent functions of the board UI
		r_app_board_ui_ctrl_rx"xxx"_esb.c	Definition of MCU-dependent functions of the board UI
	cfg	r_app_control_cfg.h	Configuration definition for the application layer
motor_module	mr_vector_rx	r_motor_mr_vector_action.c	Definition of action functions
		r_motor_mr_vector_api.c/h	Definition of API functions for the manager module
		r_motor_mr_vector_manager.c/h	Definition of local functions for the manager module
		r_motor_mr_vector_protection.c/h	Definition of functions for the protection function
		r_motor_mr_vector_ statemachine.c/h	Definition of functions related to state transition
	current_rx	r_motor_current_api.c/h	Definition of API functions for the current control module
		r_motor_current.c/h	Definition of local functions for the current control module
		r_motor_current_modulation.c/h	Definition of functions for the modulation module
		r_motor_current_volt_err_comp.lib/h	Definition of functions for the voltage error compensation module
		r_motor_current_pi_gain_calc.c	Definition of functions for calculating the control gain of the current control module
	speed_rx	r_motor_speed_api.c/h	Definition of API functions for the speed control module
		r_motor_speed.c/h	Definition of local functions for the speed control module
		r_motor_speed_fluxwkn.lib/h	Definition of functions for the magnetic flux weakening module
		r_motor_speed_extobserver.lib/h	Definition of functions for the disturbance observer module
		r_motor_speed_pi_gain_calc.c	Definition of functions for calculating the control gain of the speed control module

Folder	**Subfolder**	**File**	**Remarks**
motor_module	position_rx	r_motor_position_api.c/h	Definition of API functions for the position control module
		r_motor_position.c/h	Definition of local functions for the position control module
		r_motor_position_profiling.c/h	Definition of functions for creating the position control command value
		r_motor_position_gain_calc.c	Definition of functions for calculating the control gain of the position control module
	ipd_rx	r_motor_ipd_api.lib/h	Definition of API functions for the IPD control module
	driver_rx	r_motor_driver.c/h	Definition of functions for the driver module
	sensor_rx	r_motor_sensor_api.c/h	Definition of API functions for the sensor module
		r_motor_sensor_mr.c/h	Definition of processing functions for the sensor module
		r_motor_sensor_mr_digital.c/h	Definition of digital output and inductive sensor processing functions for the sensor module
		r_motor_sensor_mr_analog.c/h	Definition of analog output processing functions for the sensor module
		r_motor_sensor_mr_spi.c/h	Definition of SPI output processing functions for the sensor module
	general	r_motor_filter.c/h	Definition of general-purpose filter functions
		r_motor_pi_control.c/h	Definition of functions for PI control
		r_motor_common.h	Common definition
	cfg	r_motor_inverter_cfg.h	Configuration definition for the inverter
		r_motor_module_cfg.h	Configuration definition for the control module
		r_motor_targetmotor_cfg.h	Configuration definition for the motor
src	smc_gen	See the next table.	Drivers and API functions generated by the Smart Configurator

Note: The xxx portion is replaced by an MCU name. (e.g., RX72T)

The Smart Configurator can be used to generate peripheral drivers easily.

The Smart Configurator saves the setting information about the microcontrollers, peripheral functions, pin functions, and other items that are used for the project in a project file (*.scfg), and references the information saved in the file. To check the settings of the peripheral functions for the sample program, see the following file, taking the RX24T as an example:

RX24T_ESB_SPM_MR_FOC_xxx_RVyyy.scfg
(In the above file name, the "xxx" portion indicates the edition: CSP indicates the CS+ edition and E2S indicates the e^2 studio edition. The "yyy" portion indicates the revision number.)

The following table shows the configuration of the folders and files generated by the Smart Configurator.

Table 4-4 Folder and File Configurations of the Smart Configurator for the RX24T

Folder	Subfolder	2nd Subfolder	File	Remarks
src	smc_gen	Config_CMT0	Config_CMT0.c/h	Definition of functions related to CMT for the control interval
			Config_CMT0_user.c	Definition of user functions related to CMT for the control interval
		Config_MOTOR	Config_MOTOR.c/h	Definition of functions related to the Motor component
			Config_MOTOR_user.c	Definition of user functions related to the Motor component
		Config_S12AD0	Config_S12AD0.c/h	Definition of functions related to 12-bit ADC
			Config_S12AD0_user.c	Definition of user functions related to 12-bit ADC
		Config_S12AD2	Config_S12AD2.c/h	Definition of functions related to 12-bit ADC
			Config_S12AD2_user.c	Definition of user functions related to 12-bit ADC
		Config_PORT	Config_PORT.c/h	Definition of functions related to ports
			Config_PORT_user.c	Definition of user functions related to ports
		Config_RSPI0	Config_RSPI0.c/h	Definition of RSPI0-related functions for SPI output
			Config_RSPI0_user.c	Definition of RSPI0-related user functions for SPI output
		Config_DTC	Config_DTC.c/h	Definition of DTC-related functions for SPI output
			Config_DTC_user.c	Definition of DTC-related user functions for SPI output
		Config_IWDT	Config_IWDT.c/h	Definition of functions related to IWDT
			Config_IWDT_user.c	Definition of user functions related to IWDT
		Config_MTU0	Config_MTU0.c/h	Definition of functions related to MTU for speed measurement
			Config_MTU0_user.c	Definition of user functions related to MTU for speed measurement
		Config_MTU1	Config_MTU1.c/h	Definition of functions related to MTU for phase counting
			Config_MTU1_user.c	Definition of user functions related to MTU for phase counting
		Config_POE	Config_POE.c/h	Definition of functions related to POE
			Config_POE_user.c	Definition of user functions related to POE

In addition to the table above, the following four folders are automatically generated when the Smart Configurator is used:

r_bsp: This folder contains various BSP (Board Support Package) files. For details, see the "readme.txt" file in the "r_bsp" folder.

general: This folder contains various files that are shared by the Smart Configurator generation drivers.

r_config: This folder contains the configuration header files for the MCU package, clocks, interrupts, and driver initialization functions that have names in the "R_xxx_Open" pattern.

r_pincfg: This folder contains various files related to pin settings.

Notes on codes generated by the Smart Configurator

The motor component of the Smart Configurator provides a single interface that you can use to configure multiple peripheral functions (such as multi-function timer pulse unit and 12-bit A/D converter) required to drive a motor with a simple and intuitive GUI.

However, if you also use other components that generate codes related to the same peripheral function (such as a component dedicated for the 12-bit A/D converter) at the same time, the register setting might be overwritten. In this case, you can use the <configuration-name>_user.c file generated by the affected component as a countermeasure.

<Reference: For the RX72T sample program>

In the 12-bit A/D converter component initialization function “R_Config_S12AD0_Create”, AN003 is set as the target for A/D conversion. However, if the motor component initialization function “R_Config_MOTOR_Create” is called after that initialization function is set, AN003 will be removed from the A/D conversion target when AN000 and AN002 are set as A/D conversion targets. Therefore, AN003 is re-set as an A/D conversion target in “Config_MOTOR_user.c”. On the other hand, if the motor component initialization function comes before the A/D converter component initialization function, AN000 and AN002 are removed from the A/D conversion target when AN003 is set. Therefore, AN000 and AN002 are re-set as A/D conversion targets in “Config_S12AD0_user.c”.

These two re-settings are performed in the sample program so that correct settings are made regardless of the order of the initialization functions for the 12-bit A/D converter component and motor component.

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 the 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 volumes on the inverter board to drive the motor, and the RMW UI, which uses the 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 the board UI or RMW UI, and switches between these UIs.
Board UI processing	Acquires 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.

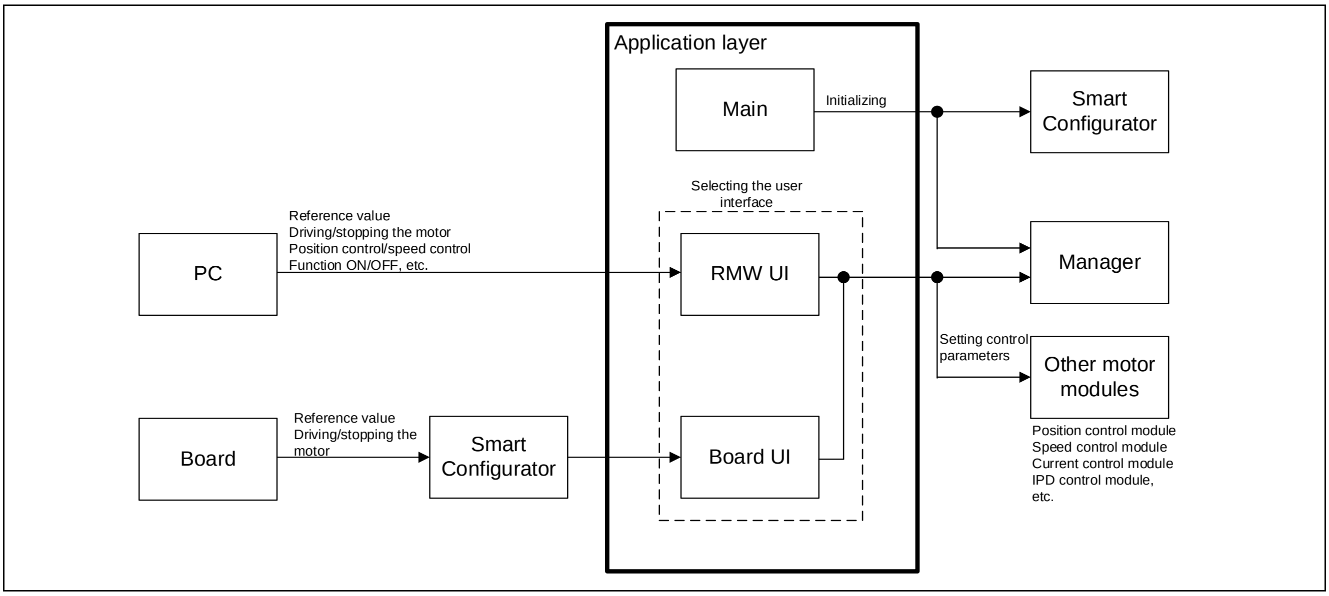


Figure 5-1 Configuration of the Application Layer

5.1.3 Flowcharts

(a) Main processing

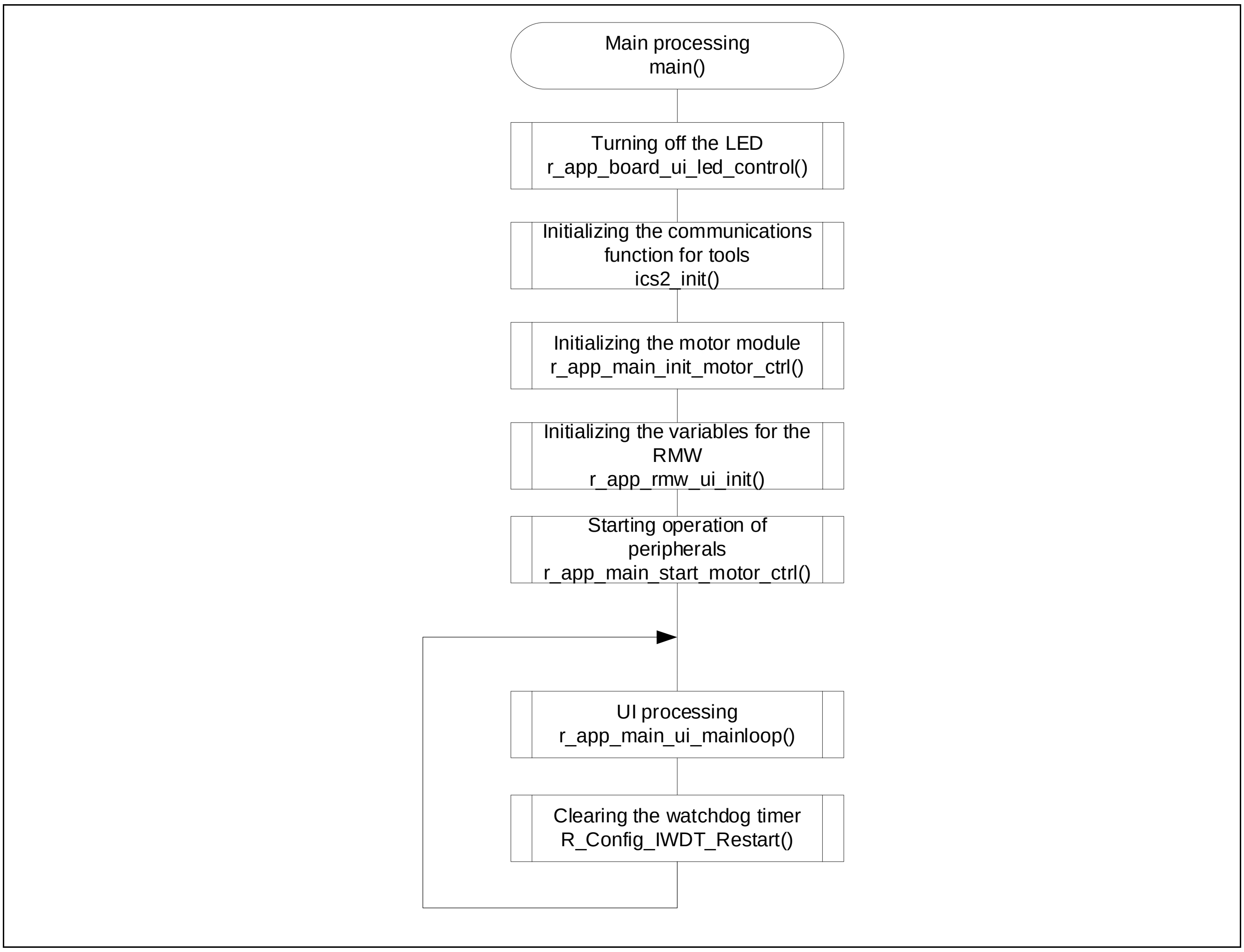


Figure 5-2 Flowchart for the Main Processing

(b) UI processing

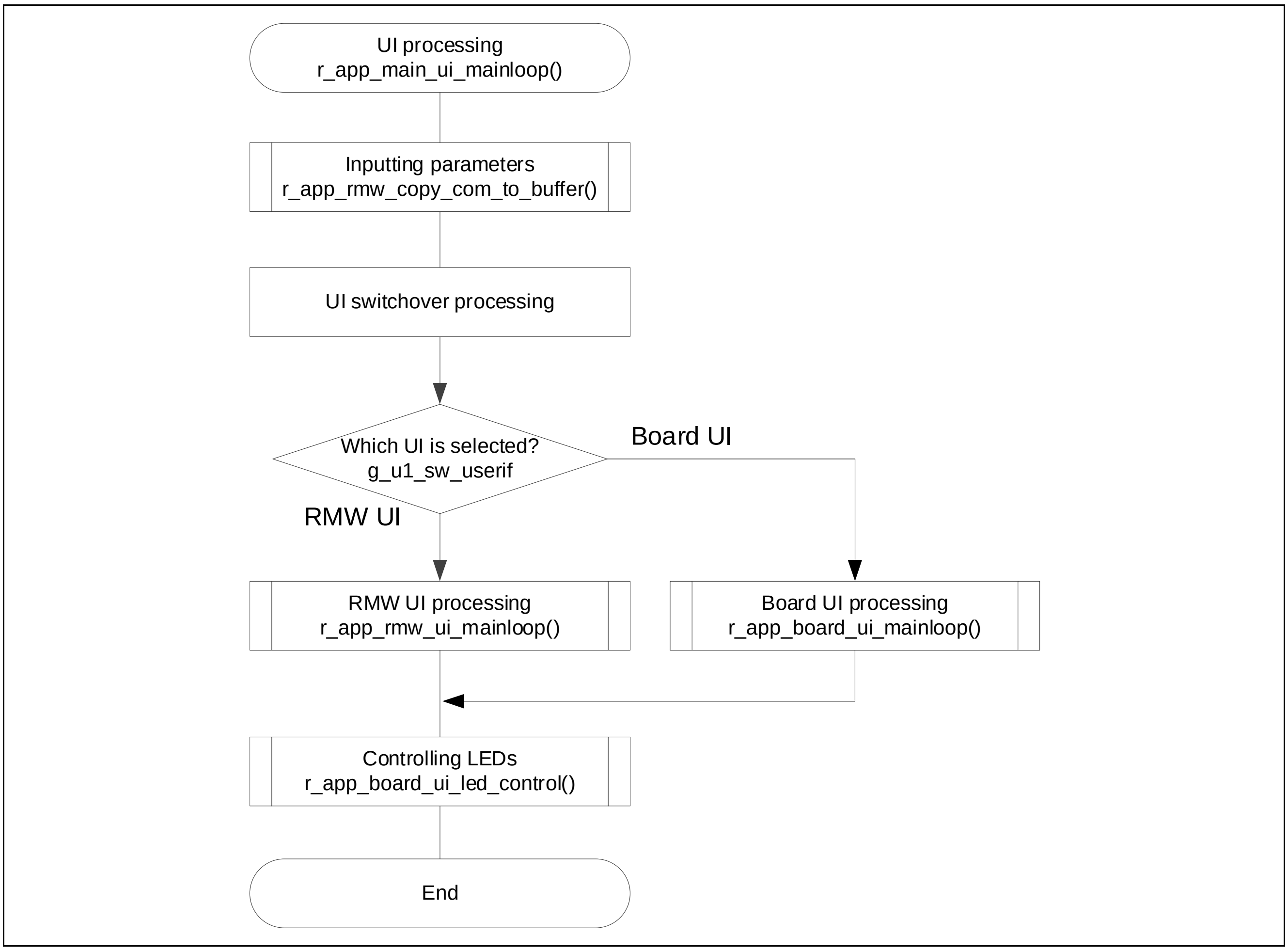


Figure 5-3 Flowchart for the UI Processing

(c) Board UI processing

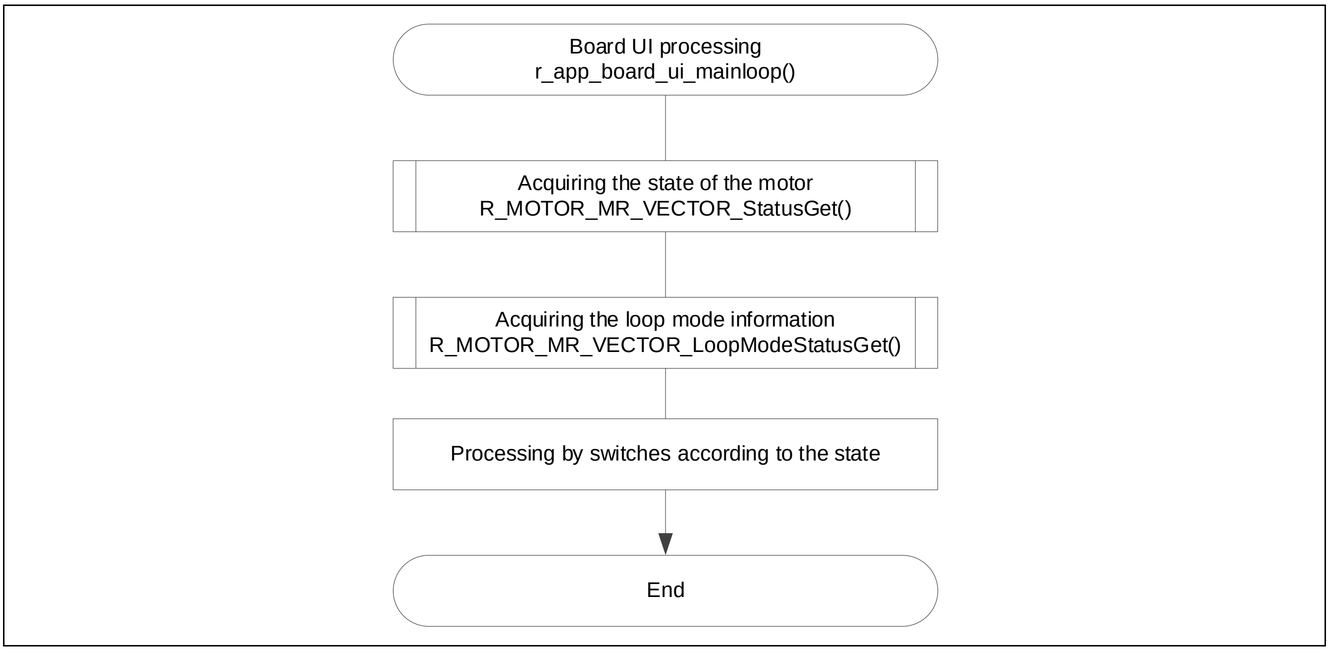


Figure 5-4 Flowchart for the Board UI Processing

(d) RMW UI processing

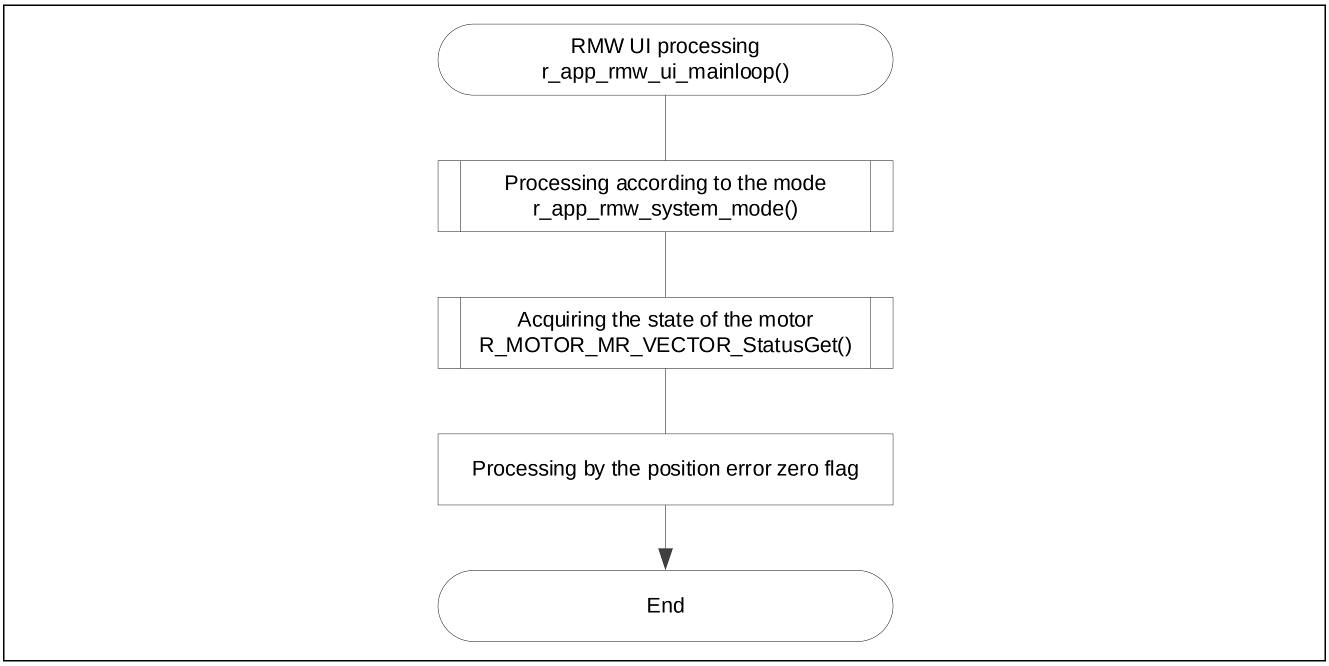


Figure 5-5 Flowchart for the RMW UI Processing

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 frequency for the current control system [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 the RMW.

Table 5-3 List of Initial Values for Configurations

Macro Name	Set Value
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	RX72M: 0x61 (SCI6) RX72T: 0x60 (SCI6) RX66T: 0x60 (SCI6) RX24T: 0x60 (SCI6) RX24U: 0x60 (SCI6) RX23T: 0x51 (SCI5) RX13T: 0x10 (SCI1)

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 the 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: Stopping the motor 1: Driving the motor 2: Error
com_u1_enable_write	Enables rewriting of variables for user entry
g_u1_enable_write	Enables rewriting 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 setting
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_mr_angle_adj_mode	Magnetic pole position detection mode 0: Position detection by forced excitation 1: Position detection using a Hall sensor
com_u2_mr_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	Ramping-up time of the d-axis current command value setting
com_f4_current_omega_hz	Natural frequency for the current control system [Hz]
com_f4_current_zeta	Damping ratio for the current control system
com_f4_speed_omega_hz	Natural frequency for the speed control system [Hz]
com_f4_speed_zeta	Damping ratio for the speed control system
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	Speed limit change rate [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 wave response 2: Trapezoidal wave response
com_u2_pos_interval_time	Stationary wait time for position response
com_u2_pos_dead_band	Dead band (number of digital output pulses) [pulses]
com_u2_pos_band_limit	Position error zero range [pulses]
com_f4_pos_omega_hz	Natural frequency for the position control system [Hz]
com_f4_pos_ff_ratio	Location 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 the 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 position profiling (mechanical angle) [rpm]
com_u1_flag_ipd_use	IPD control module setting 0: Disabled 1: Enabled
com_f4_ipd_speed_k_ratio	Speed gain ratio during IPD control
com_f4_ipd_pos_kp_ratio	P control amount ratio for the position during IPD control
com_f4_ipd_omega_hz	Natural frequency for IPD control
com_f4_ipd_pos_ff_ratio	IPD control feed-forward gain
com_u1_flag_volt_err_comp_use	Voltage error compensation setting 0: Disabled 1: Enabled
com_u1_flag_fluxwkn_use	Flux weakening control setting 0: Disabled 1: Enabled
s_u1_cnt_ics	Counter for counting the number of times an ICS watchpoint is skipped

Table 5-5 List of Variables of the Structure for the 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 setting
	st_motor_parameter_t	Structure for the motor parameters
	f4_max_speed_rpm	Maximum speed [rpm]
	u1_ctrl_loop_mode	Control loop mode (position control and speed control)
	u1_mr_angle_adj_mode	Selection of the initial position detection mode
	u2_mr_cpr	Number of pulses per rotation of the digital output magnet sensor [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 the current control system [Hz]
	f4_current_zeta	Damping ratio for the current control system
	f4_speed_omega_hz	Natural frequency for the speed control system [Hz]
	f4_speed_zeta	Damping ratio for the speed control system
	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 state
	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 the position control system [Hz]
	f4_pos_ff_ratio	Location 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 the speed control module observer [Hz]
	f4_accel_time	Acceleration time [s]

Structure	Variable	Description
st_rmw_param_buffer_t Structure for updating the RMW variables	f4_posprof_max_speed_rpm	Maximum speed value for position profiling (mechanical angle) [rpm]
	u1_flag_ipd_use	Flag for whether to use IPD control
	f4_ipd_speed_k_ratio	Speed constant for IPD control
	f4_ipd_pos_kp_ratio	Position kp constant for IPD control
	f4_ipd_omega_hz	IPD control frequency [Hz]
	f4_ipd_pos_ff_ratio	Position feed-forward gain for IPD control
	u1_flag_volt_err_comp_use	Flag for whether to use voltage error compensation
	u1_flag_fluxwkn_use	Flag for whether to use 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_SW_ON	0	The switch is on.
	BOARD_SW_OFF	1	The switch 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	50	Speed margin for VR1 [rpm]
	BOARD_VR1_SCALING_POS	(180 + 1) / (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	0x61	Selection of the SCI channel to be used by users
	APP_CFG_USE_UI	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	5	RMW watchpoint skip count
	ICS_INT_LEVEL	6	RMW interrupt priority
	ICS_BRR	250	Communications baud rate for the RMW
	ICS_INT_MODE	1	Communications mode selection for the RMW
	ICS_SCI_CH_SELECT	CFG_SCI_CH_SELECT	SCI channel to be used

Note: In ICS2_RXxxx.h, a macro that defines the channel used for communications via RMW is provided for each MCU. The xxx portion is replaced with the name of the MCU.

Table 5-7 List of Macro Definitions

<table>
<tr><th>Macro name</th><th>RX13T</th><th>RX23T</th><th>RX24T, RX24U</th><th>RX66T</th><th>RX72T</th><th>RX72M</th></tr>
<tr><td>ICS_DECIMATION</td><td colspan="3">3</td><td colspan="3">5</td></tr>
<tr><td>ICS_INT_LEVEL</td><td colspan="6">6</td></tr>
<tr><td>ICS_BRR</td><td>3</td><td colspan="3">4</td><td colspan="2">250</td></tr>
<tr><td>ICS_INT_MODE</td><td colspan="6">1</td></tr>
</table>

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, configure and update them by using the RMW. For details about how to use the RMW, see 3, Quick Start Guide and the Renesas Motor Workbench V.2.00 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 Functionality

Table 5-8 lists the functions of the manager module.

Table 5-8 List of Manager Module Functions

Function	Description
Mode management	Switches the operating 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 the Smart Configurator.

5.2.2 Module Configuration Diagram

Figure 5-6 shows the module configuration.

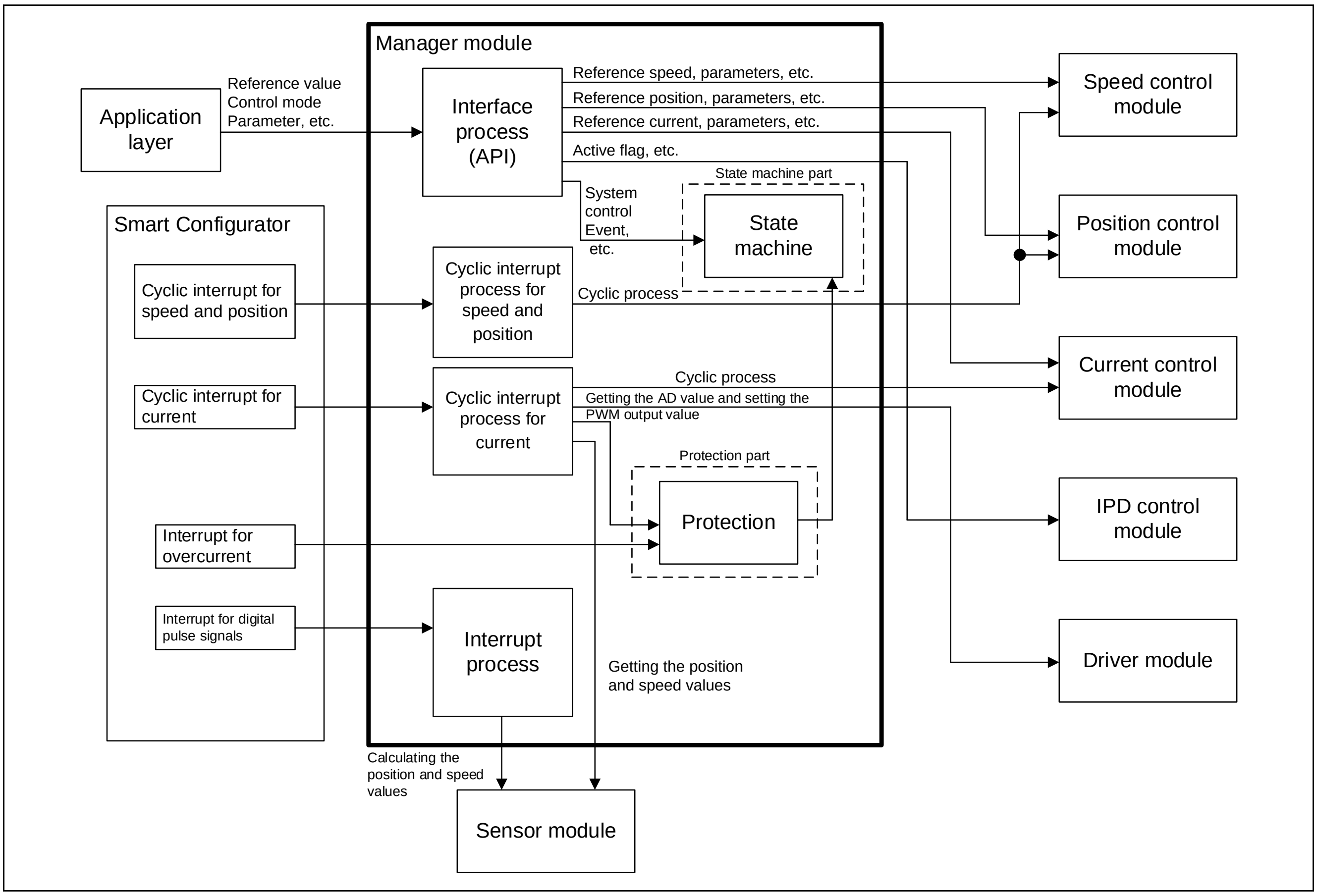
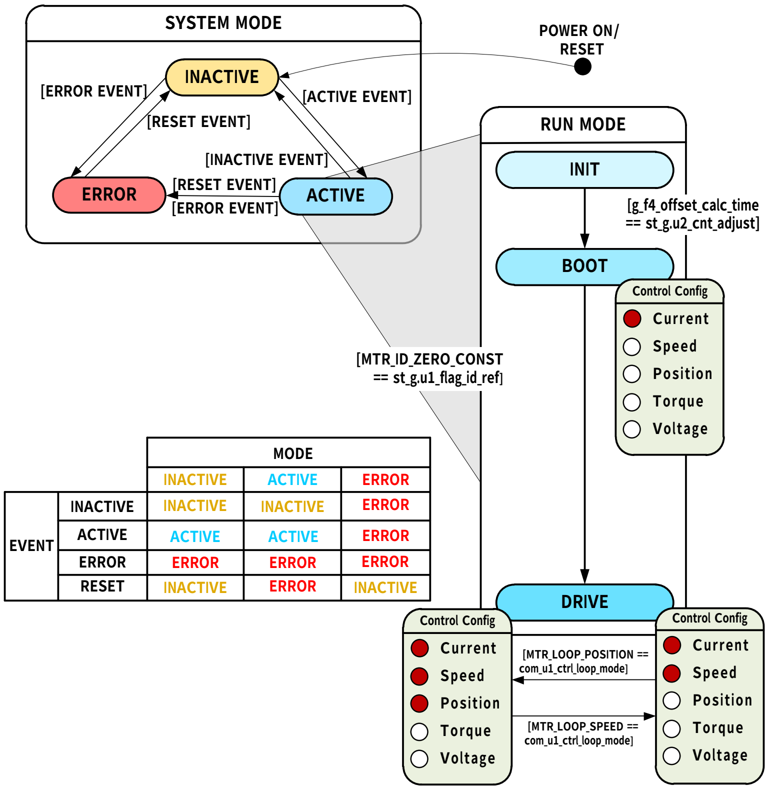


Figure 5-6 Manager Module Configuration Diagram

5.2.3 Mode Management

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



		MODE		
		INACTIVE	ACTIVE	ERROR
EVENT	INACTIVE	INACTIVE	INACTIVE	ERROR
	ACTIVE	ACTIVE	ACTIVE	ERROR
	ERROR	ERROR	ERROR	ERROR
	RESET	INACTIVE	ERROR	INACTIVE

Figure 5-7 State Transition Diagram for the Magnet Sensor Vector Control Software

(1) System mode

This mode is used to indicate the system operating state. The system enters a different state in response to the input of an event signal. There are three system operating states: INACTIVE (the motor is stopped), ACTIVE (the motor is running), and ERROR (an error has occurred).

(2) Run mode

This mode is 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 how the system operating 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 5-9 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 5-10.

- Overcurrent error

  Overcurrent errors can be detected on the hardware and in the software.

  The PWM outut pin is placed in the high-impedance state in response to an emergency stop signal (overcurrent detection) from the hardware.

  This function monitors U-, V-, and W-phase currents at the overcurrent monitoring interval. When this function detects an overcurrent (the state 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 (MP_NOMINAL_CURRENT_RMS).

- Overvoltage error

  This function monitors the inverter bus voltage at the overvoltage monitoring interval. When the function detects an overvoltage (that is, a voltage above the overvoltage limit value), it brings the program to an emergency stop. The overvoltage limit value is preset in consideration of conditions such as an 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 (that is, a voltage below the low-voltage limit value), it brings the program to an emergency stop. The low-voltage limit value is preset in consideration of conditions such as an 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 5-10 Values Specified for the System Protection Function Settings

Overcurrent error	Overcurrent limit value [A]	3.82
	Monitoring interval [μs]	Current control interval *[1]
Overvoltage error	Overvoltage limit value [V]	28
	Monitoring interval [μs]	Current control interval *[1]
Low-voltage error	Low-voltage limit value [V]	14
	Monitoring interval [μs]	Current control interval *[1]
Rotation speed error	Speed limit value [rpm]	4500
	Monitoring interval [μs]	Current control interval *[1]

Note: 1. See Table 4-1 Basic Specifications of Magnet Sensor Vector Control Software.

5.2.5 Flowcharts

The manager module performs processing in response to the occurrences of interrupts that are set by 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.

(a) Interrupt processing for current control

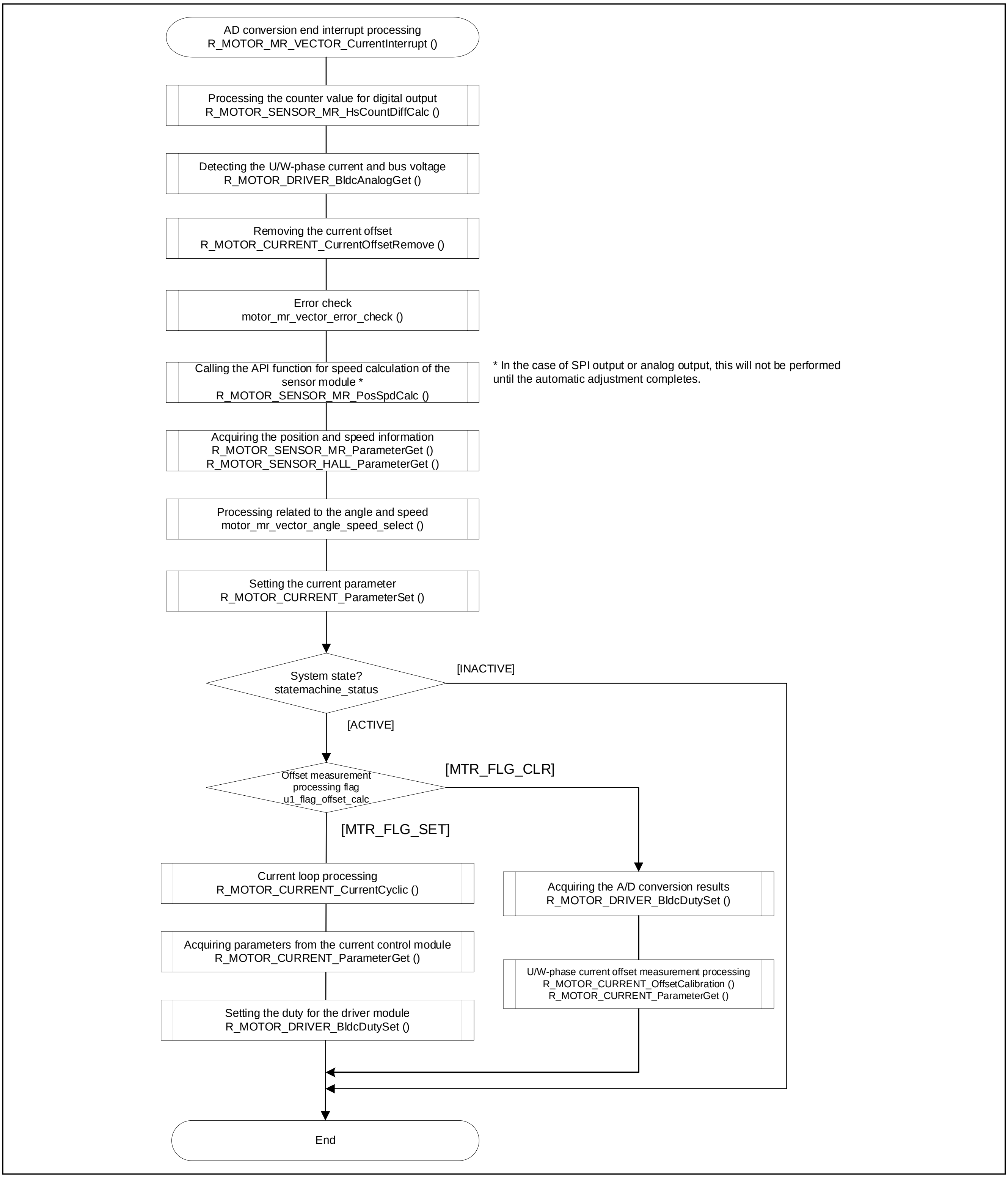


Figure 5-8 Interrupt Processing Flowchart for Current Control

(b) Interrupt processing for position and speed control

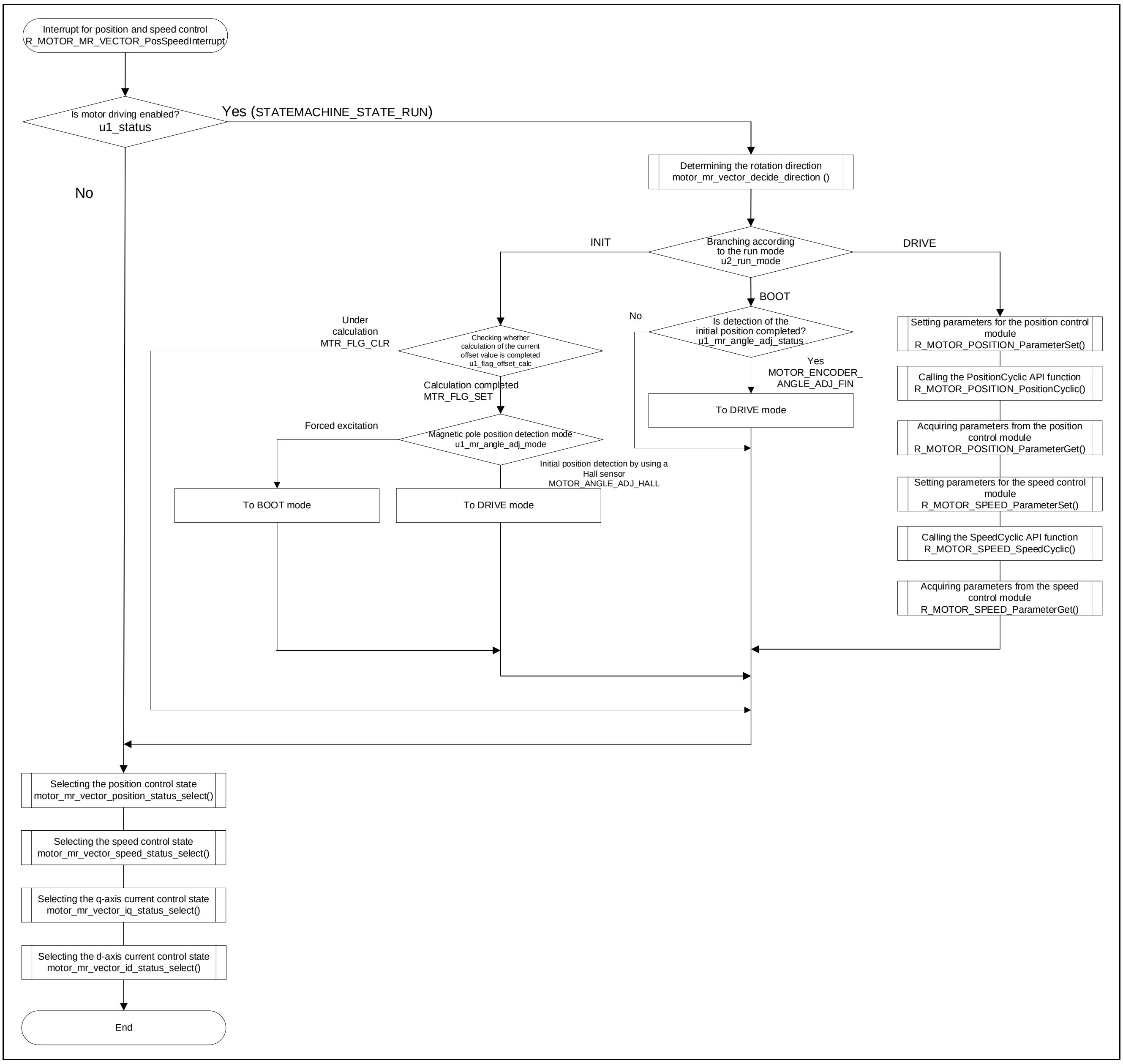


Figure 5-9 Interrupt Processing Flowchart for Position and Speed Control (Digital Output)

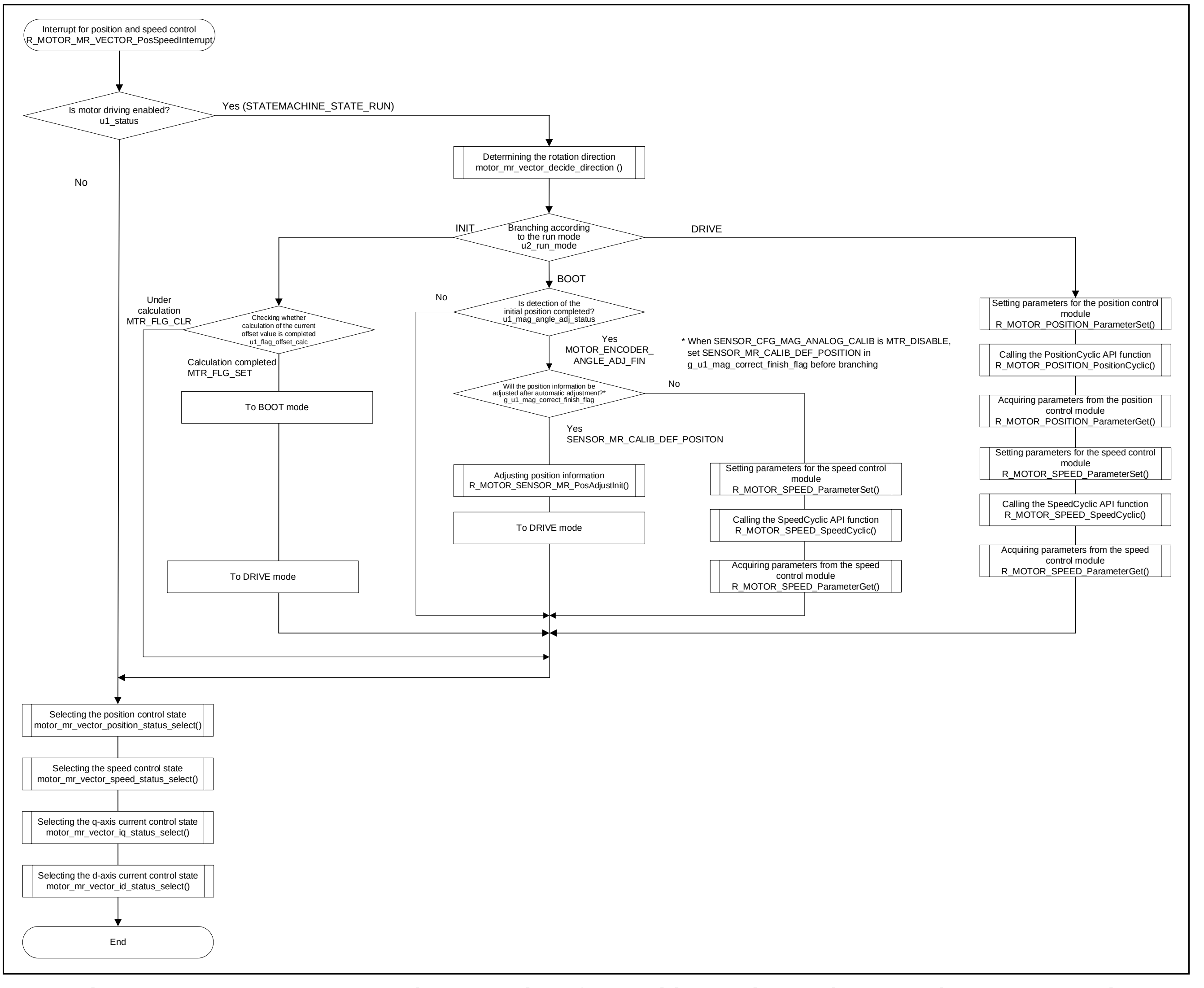


Figure 5-10 Interrupt Processing Flowchart for Position and Speed Control (Analog Output)

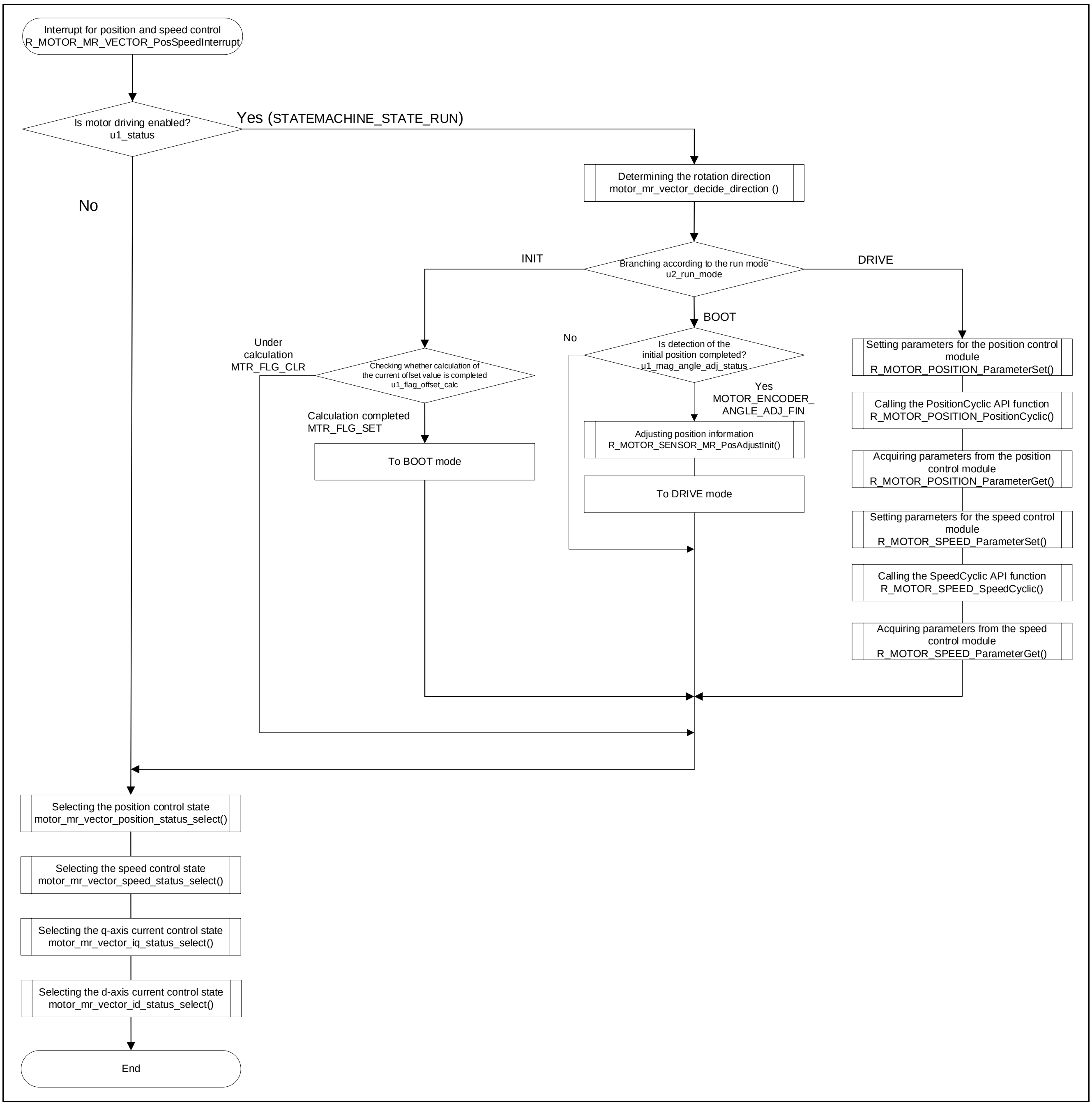


Figure 5-11 Interrupt Processing Flowchart for Position and Speed Control (SPI Output)

(c) Overcurrent detection interrupt processing

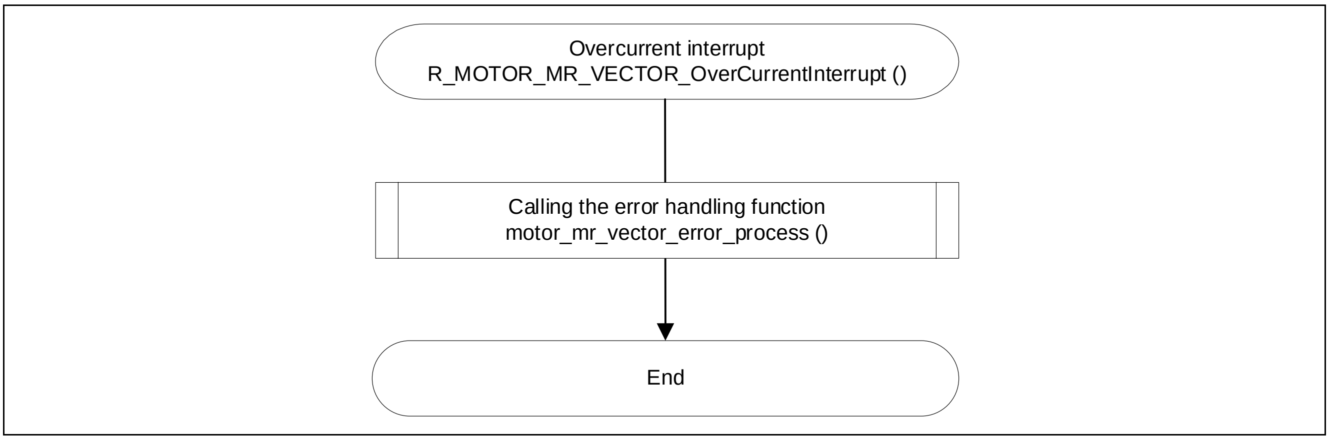


Figure 5-12 Overcurrent Detection Interrupt Processing Flowchart

(d) Interrupt processing for digital pulse signals

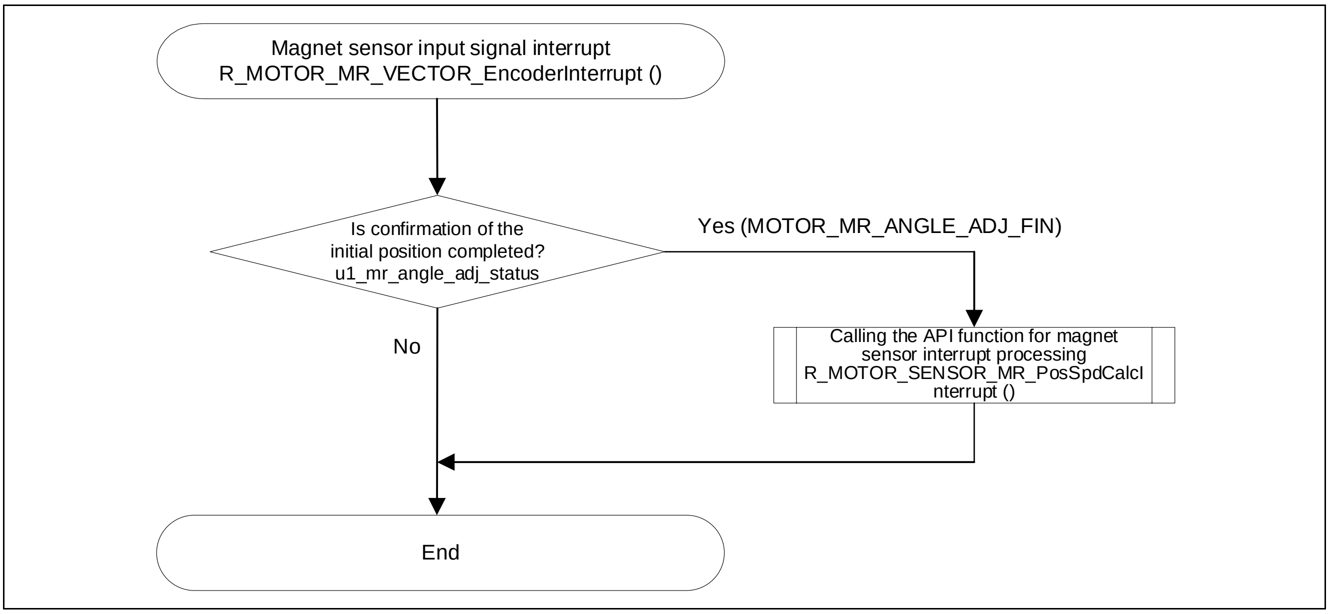


Figure 5-13 Digital Pulse Signal Counting Capture Interrupt Processing Flowchart

5.2.6 API

Table 5-11 lists the manager module API functions.

Table 5-11 List of API Functions

API Function	Description
R_MOTOR_MR_VECTOR_Open	Generates an instance of the manager module. This function also uses the Open function of the control module to generate an instance.
R_MOTOR_MR_VECTOR_Close	Places the modules, including the manager module, in the reset state.
R_MOTOR_MR_VECTOR_Reset	Initializes the modules, including the manager module.
R_MOTOR_MR_VECTOR_ParameterUpdate	Updates the control parameter settings of this module. This function also updates the control parameters for the related modules.
R_MOTOR_MR_VECTOR_MotorStart	Places the motor in the running state.
R_MOTOR_MR_VECTOR_MotorStop	Places the motor in the stopped state.
R_MOTOR_MR_VECTOR_MotorReset	Releases the system from the error state.
R_MOTOR_MR_VECTOR_ErrorSet	Places the system in the error state.
R_MOTOR_MR_VECTOR_PositionSet	Sets the position command value. This function is enabled when position control is applied.
R_MOTOR_MR_VECTOR_PositionGet	Acquires the position information.
R_MOTOR_MR_VECTOR_SpeedSet	Sets the speed command value. This function is enabled when speed control is applied.
R_MOTOR_MR_VECTOR_SpeedGet	Acquires the speed information.
R_MOTOR_MR_VECTOR_StatusGet	Acquires the state from the state machine.
R_MOTOR_MR_VECTOR_ErrorStatusGet	Acquires the error state.
R_MOTOR_MR_VECTOR_CtrlTypeSet	Sets the control method. To change the control method, place the motor in the stopped state. 0: Position control 1: Speed control
R_MOTOR_MR_VECTOR_LoopModeStatusGet	Acquires the control method. 0: Position control 1: Speed control
R_MOTOR_MR_VECTOR_PositionCommandModeSet	Selects the mode of position command generation for position control. To apply position control, set 1 or 2. 0: The position is always 0. 1: Step wave response operation 2: Trapezoidal driving operation
R_MOTOR_MR_VECTOR_InPositionFlagGet	Acquires the position control completion state. This function is enabled when position control is applied. 0: Position control is not completed yet. 1: Position control has been completed.
R_MOTOR_MR_VECTOR_PosSpeedInterrupt	Performs interrupt processing for position control and speed control.
R_MOTOR_MR_VECTOR_CurrentInterrupt	Performs interrupt processing for current control.

API Function	Description
R_MOTOR_MR_VECTOR_OverCurrentInterrupt	Performs interrupt processing when an overcurrent occurs.
R_MOTOR_MR_VECTOR_DigitalPulseInterrupt	Performs interrupt processing for digital output pulse signals.
R_MOTOR_MR_VECTOR_IPDEnableSet	Enables or disables the IPD control module. 0: Disabled 1: Enabled

5.2.7 Configurations

Table 5-12 lists the configurations for the manager module.

Table 5-12 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	ADC operating frequency [MHz]
	MOTOR_MCU_CFG_AD_SAMPLING_CYCLE	ADC sampling state [cycles]
	MOTOR_MCU_CFG_AD12BIT_DATA	ADC resolution
	MOTOR_MCU_CFG_ADC_OFFSET	ADC intermediate data
	MOTOR_TYPE_BLDC	Motor used (BLDC)
	MOTOR_COMMON_CFG_LOOP_MODE	Sets the default run mode.
	MOTOR_COMMON_CFG_IPD_CTRL	Sets the default IPD control usage.
	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: √3 STM: √2
	MOTOR_MCU_CFG_TFU_OPTIMIZE	Sets the processing of TFU-specific functions MTR_ENABLE MTR_DISABLE

Table 5-13 List of Initial Values for Configurations

Macro Name	RX13T	RX23T	RX24T, RX24U	RX66T	RX72T	RX72M
MOTOR_MCU_CFG_PWM_TIMER_FREQ	32	40	80	160	200	120
MOTOR_MCU_CFG_CARRIER_FREQ	16	20				
MOTOR_MCU_CFG_INTR_DECIMATION	1			0		
MOTOR_MCU_CFG_AD_FREQ	32	40	40	40	50	60.0
MOTOR_MCU_CFG_AD_SAMPLING_CYCLE	47	47	47	45	45	45
MOTOR_MCU_CFG_AD12BIT_DATA	4095.0f					
MOTOR_MCU_CFG_ADC_OFFSET	0x7FF					
MOTOR_TYPE_BLDC	Defined					
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_DISABLE				MTR_ENABLE	

5.2.8 Structure and Variable Information

Table 5-14 lists the structures and variables for the manager module. For the manager module, the structure for the manager module (g_st_mr_vector) is defined by securing an instance of the module from the API.

Table 5-14 List of Structures and Variables

Structure	Variable	Description
st_mr_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_mag_angle_adj_mode	Selection of the initial position detection mode 0: Forced excitation 1: Using a Hall sensor (unused)
	u1_mag_angle_adj_status	State of detection of the initial position
	u2_mag_angle_adj_time	Wait time for detection of the initial position [µs]
	u2_mag_angle_adj_cnt	Counting of the wait time for detection of the initial position
	u1_flag_ipd_use	Flag for whether to use IPD control 0: IPD control disabled 1: IPD control enabled
	u2_error_status	Error state
	u2_run_mode	Run mode 0: Initialization 1: Startup preparation 2: Driving the motor
	u1_state_id_ref	State of the d-axis current command value
	u1_state_iq_ref	State of the q-axis current command value
	u1_state_speed_ref	State of the speed command value
	u2_mag_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_ctrl_period	Current loop control interval [s]

Structure	Variable	Description
st_mr_vector_control_t Structure for the manager module	f4_max_speed_rc_fil_rpm	Maximum speed (rc filter used)
	st_current_output	Structure for output from the current control module
	st_speed_output	Structure for output from the speed control module
	st_position_output	Structure for output from the position control module
	st_sensor_mr_output	Structure for encoder output of the sensor module
	st_stm	Structure for the state machine
	st_motor	Motor parameter structure
	p_st_cc	Pointer to the structure for the current control module
	p_st_sc	Pointer to the structure for the speed control module
	p_st_pc	Pointer to the structure for the position control module
	p_st_ipd	Pointer to the structure for the IPD control module
	p_st_sensor	Pointer to the structure for the sensor module
	p_st_driver	Pointer to the structure for the driver module
st_mr_vector_cfg_t Structure for setting the manager module control parameters	u1_mag_angle_adj_mode	Selection of the initial position detection mode
	u2_mag_cpr	Number of digital output pulses per rotation
	f4_nominal_current_rms	Current limit value [A]
	f4_overspeed_limit_rpm	Speed limit value [rpm]
	st_motor	Motor parameter structure

Table 5-15 List of Variables for Automatic Adjustment of Analog Output Signals

Variable	Set Value	Description
g_u1_mag_correct_finish_flag	SENSOR_MR_CALIB_PREPARED	Automatic adjustment complete flag
g_u1_mag_open_start_trig	0	Open-loop operation start trigger
g_u1_mag_open_start_flag	SENSOR_MR_OPENLOOP_INACTIVE	Open-loop operation flag
s_u1_mag_speed_calc_counter	0	Counter value for speed calculation cycles
s_u1_mag_sens_cnt	0	Angle acquisition counter during automatic adjustment
g_u1_mag_calib_status	SENSOR_MR_CALIB_INIT_SEQ	State during automatic adjustment
g_u1_mag_calib_cnt	0	Number of times to acquire automatic adjustment data
s_u1_mag_get_signal_max_diff	0	Difference between maximum sensor output values
s_u1_mag_save_status_speed	0	Holds the state of the speed during automatic adjustment
s_u1_mag_save_status_id	0	Holds the state of the d-axis current during automatic adjustment

Variable	Set Value	Description
g_u2_mag_get_adc_cnt	0	Number of times to acquire AD during automatic adjustment
s_f4_mag_sin_max1	0.0f	Maximum value of the sine wave at the first automatic adjustment [V]
s_f4_mag_sin_max2	0.0f	Maximum value of the sine wave at the second automatic adjustment [V]
s_f4_mag_sin_max_ave	0.0f	Average value of the maximum sine wave values at the automatic adjustments [V]
s_f4_mag_sin_min1	0.0f	Minimum value of the sine wave at the first automatic adjustment [V]
s_f4_mag_sin_min2	0.0f	Minimum value of the sine wave at the second automatic adjustment [V]
s_f4_mag_sin_min_ave	0.0f	Average value of the maximum sine wave values at the automatic adjustments [V]
s_f4_mag_sin_integral1	0.0f	Integral value of the sine wave data acquired at the first automatic adjustment [V]
s_f4_mag_sin_integral2	0.0f	Integral value of the sine wave data acquired at the second automatic adjustment [V]
s_f4_mag_sin_integral_ave	0.0f	Average value of the integral values of the sine wave acquired at the automatic adjustments
s_f4_mag_cos_max1	0.0f	Maximum value of the cosine wave at the first automatic adjustment [V]
s_f4_mag_cos_max2	0.0f	Maximum value of the cosine wave at the second automatic adjustment [V]
s_f4_mag_cos_max_ave	0.0f	Average value of the maximum cosine wave values acquired at the automatic adjustments [V]
s_f4_mag_cos_min1	0.0f	Minimum value of the cosine wave at the first automatic adjustment [V]
s_f4_mag_cos_min2	0.0f	Minimum value of the cosine wave at the second automatic adjustment [V]
s_f4_mag_cos_min_ave	0.0f	Average value of the maximum cosine wave values acquired at the automatic adjustments [V]
s_f4_mag_cos_integral1	0.0f	Integral value of the cosine wave data acquired at the first automatic adjustment [V]
s_f4_mag_cos_integral2	0.0f	Integral value of the cosine wave data acquired at the second automatic adjustment [V]
s_f4_mag_cos_integral_ave	0.0f	Average value of the integral values of the cosine wave acquired at the automatic adjustments [V]
s_f4_mag_sin_max	0.0f	Maximum value of the sine wave [V]
s_f4_mag_cos_max	0.0f	Maximum value of the cosine wave [V]
s_f4_mag_sin_min	0.0f	Minimum value of the sine wave [V]
s_f4_mag_cos_min	0.0f	Minimum value of the cosine wave [V]
s_f4_mag_sin_integral	0.0f	Integral value of the sine wave [V]
s_f4_mag_cos_integral	0.0f	Integral value of the cosine wave [V]
s_f4_mag_add_cos_sin_max	0.0f	Maximum value of the cosine wave + sine wave [V]
s_f4_mag_add_cos_sin_min	0.0f	Minimum value of the cosine wave + sine wave [V]

Variable	Set Value	Description
s_f4_mag_sub_cos_sin_max	0.0f	Maximum value of the cosine wave - sine wave [V]
s_f4_mag_sub_cos_sin_min	0.0f	Minimum value of the cosine wave - sine wave [V]
s_f4_mag_sin_offset	0.0f	Offset value for the sine wave [V]
s_f4_mag_cos_offset	0.0f	Offset value for the cosine signal [V]
s_f4_mag_sin_gain	0.0f	Gain correction value for the sine wave
s_f4_mag_cos_gain	0.0f	Gain correction value for the cosine wave
s_f4_mag_add_cos_sin_gain	0.0f	Gain correction value for the cosine wave + sine wave
s_f4_mag_sub_cos_sin_gain	0.0f	Gain correction value for the cosine wave - sine wave
s_f4_mag_angle_openloop	0.0f	Open-loop angle during automatic adjustment [rad]
s_f4_mag_angle_openloop_pre	0.0f	Previous open-loop angle during automatic adjustment [rad]
s_f4_mag_speed_openloop	0.0f	Open-loop speed during automatic adjustment [rad/s]
s_f4_mag_speed_openloop_pre	0.0f	Previous open-loop speed during automatic adjustment [rad/s]

5.2.9 Macro Definition

Table 5-16 lists the macros for the manager module.

Table 5-16 List of Macros

File Name	Macro Name	Defined Value	Remarks
r_motor_mr_vector_api.h	MOTOR_LOOP_POSITION	0	Position control mode
	MOTOR_LOOP_SPEED	1	Speed control mode
	MOTOR_MR_VECTOR_ERROR_NONE	(0x0000)	An error state. There is no error.
	MOTOR_MR_VECTOR_ERROR_OVER_CURRENT_HW	(0x0001)	An error state. A hardware overcurrent error has occurred.
	MOTOR_MR_VECTOR_ERROR_OVER_VOLTAGE	(0x0002)	An error state. An overvoltage error has occurred.
	MOTOR_MR_VECTOR_ERROR_OVER_SPEED	(0x0004)	An error state. An overspeed error has occurred.
	MOTOR_MR_VECTOR_ERROR_HALL_PATTERN	(0x0020)	An error state. A Hall pattern error has occurred.
	MOTOR_MR_VECTOR_ERROR_LOW_VOLTAGE	(0x0080)	An error state. A low-voltage error has occurred.
	MOTOR_MR_VECTOR_ERROR_OVER_CURRENT_SW	(0x0100)	An error state. A software overcurrent error has occurred.
	MOTOR_MR_VECTOR_ERROR_UNKNOWN	(0xffff)	An error state. 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
r_motor_mr_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_MR_ANGLE_ADJ_90DEG	0	State at startup. The encoder is positioned at 90 degrees.
	MOTOR_MR_ANGLE_ADJ_0DEG	1	State at startup. The encoder is positioned at 0 degrees.
	MOTOR_MR_ANGLE_ADJ_FIN	2	State at startup. Determination of the initial position is completed.
r_motor_mr_vector_api.h	MOTOR_CTRL_TYPE_POS	0	Macro for switching the control method. Position control mode.
	MOTOR_CTRL_TYPE_SPEED	1	Macro for switching the control method. Speed control mode.

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 5-17 shows the values set in the sample program.

Table 5-17 Motor and Inverter Parameter Settings

File Name	Macro Name	Set Value	Description
r_motor_inverter_cfg.h	INVERTER_CFG_SHUNT_RESIST	0.010f	Shunt resistance value [ohm]
	INVERTER_CFG_DEADTIME	2.0f	Dead time [µs]
	INVERTER_CFG_VOLTAGE_GAIN	22.3f	Coefficient for voltage detection
	INVERTER_CFG_CURRENT_AMP_GAIN	20.0f/5.0f * 3.3f	Gain of the amplifier for current detection
	INVERTER_CFG_CURRENT_LIMIT	10.0f	Overcurrent limit value for the inverter board [A]
	INVERTER_CFG_OVERVOLTAGE_LIMIT	28.0f	Overvoltage limit [V]
	INVERTER_CFG_UNDERVOLTAGE_LIMIT	14.0f	Low-voltage limit [V]
	INVERTER_CFG_INPUT_V	24.0f	Input voltage [V]
	INVERTER_CFG_ADC_REF_VOLTAGE	3.3f *2	Analog power supply voltage for the microcontroller [V]
	INVERTER_CFG_COMP_V0	0.672f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V1	0.945f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V2	1.054f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V3	1.109f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_V4	1.192f	Coefficient for compensation of the voltage error [V] *1
	INVERTER_CFG_COMP_I0	0.013f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I1	0.049f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I2	0.080f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I3	0.184f	Coefficient for compensation of the voltage error [A] *1
	INVERTER_CFG_COMP_I4	0.751f	Coefficient for compensation of the voltage error [A] *1

File Name	Macro Name	Set Value	Description
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]

Notes: 1. For details, see 5.5, Voltage Error Compensation (Current Control Module).

2. Make sure that the value is compatible with the specifications of the CPU card that is used. For example, for an RXxxT card, set 5.0f. The xxx portion is replaced with the name of the MCU.

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 5-14 and Figure 5-15 show starting up of the motor.

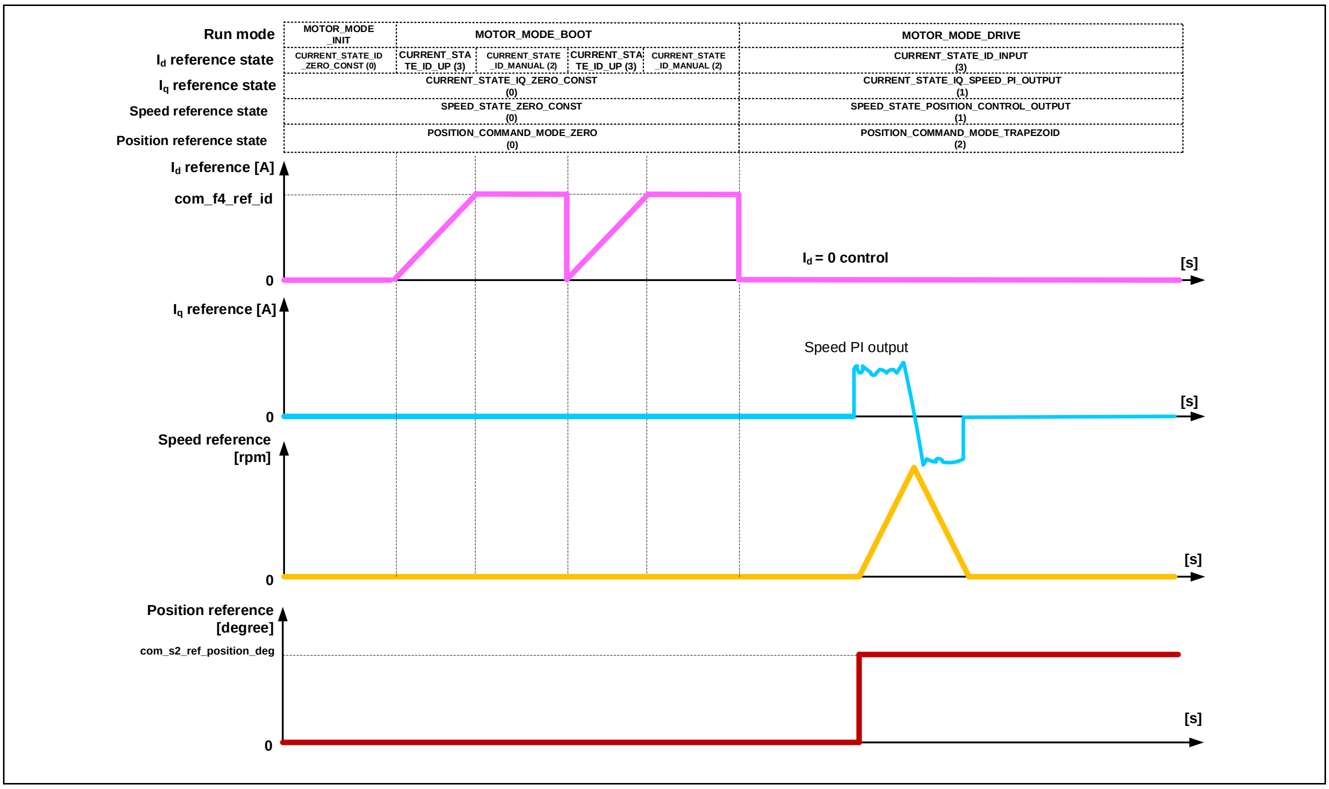


Figure 5-14 Motor Start Control Based on Vector Control Using a Magnet Sensor (in the Case of Position Control)

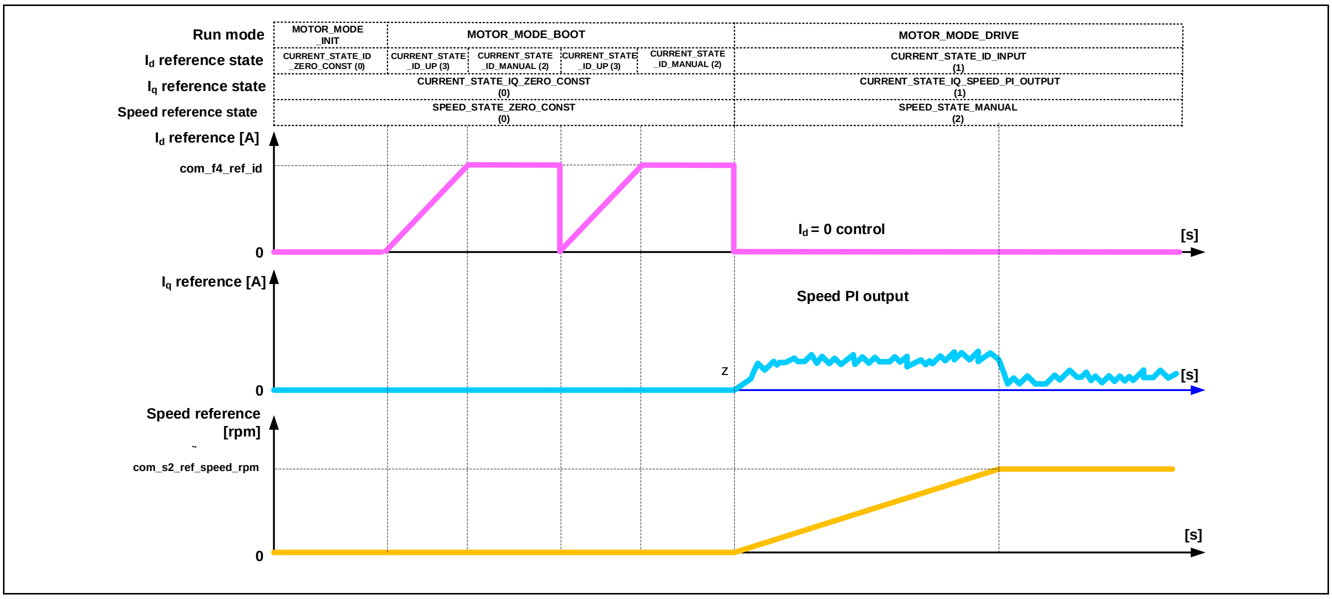


Figure 5-15 Motor Start Control Based on Vector Control Using a Magnet Sensor (in the Case of Speed Control)

(a) 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 5-16 to pull in the magnet so that the directions of the d axis and current vector match. Figure 5-17 shows the startup sequence that applies in this case.

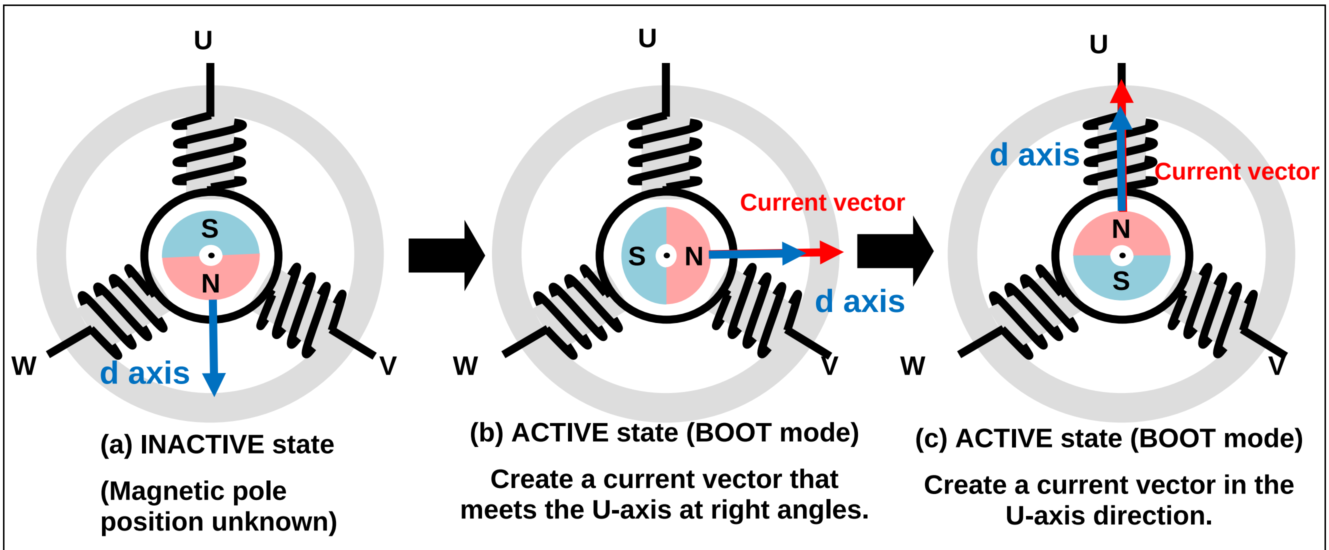


Figure 5-16 Determining the Position of a Permanent Magnet

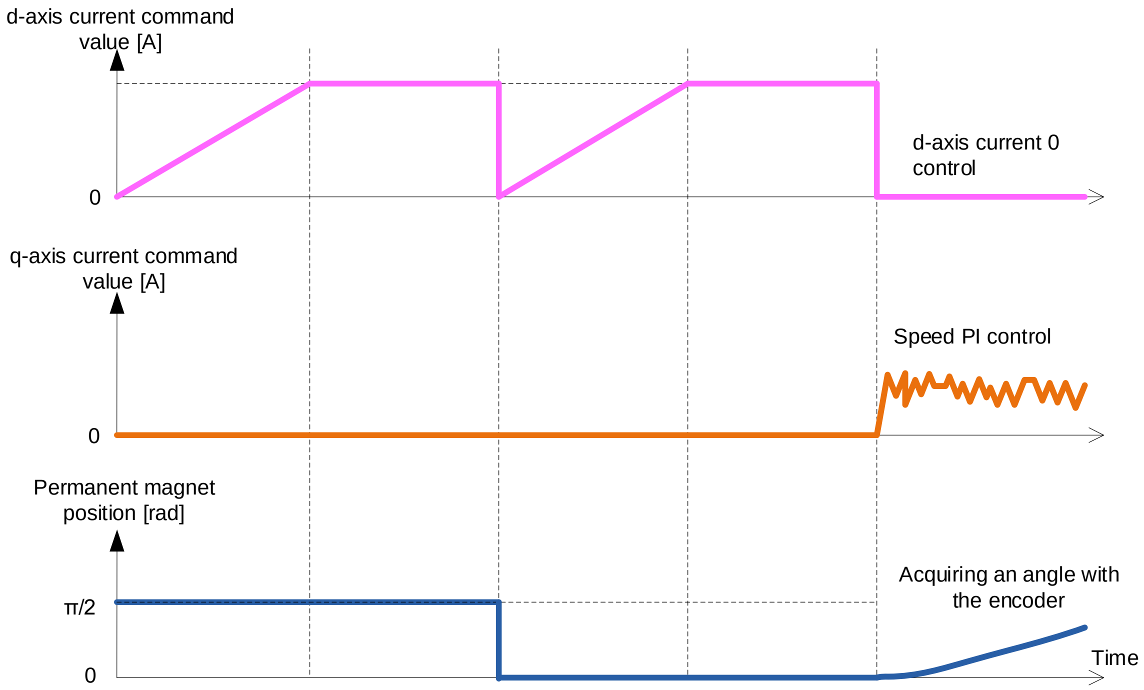
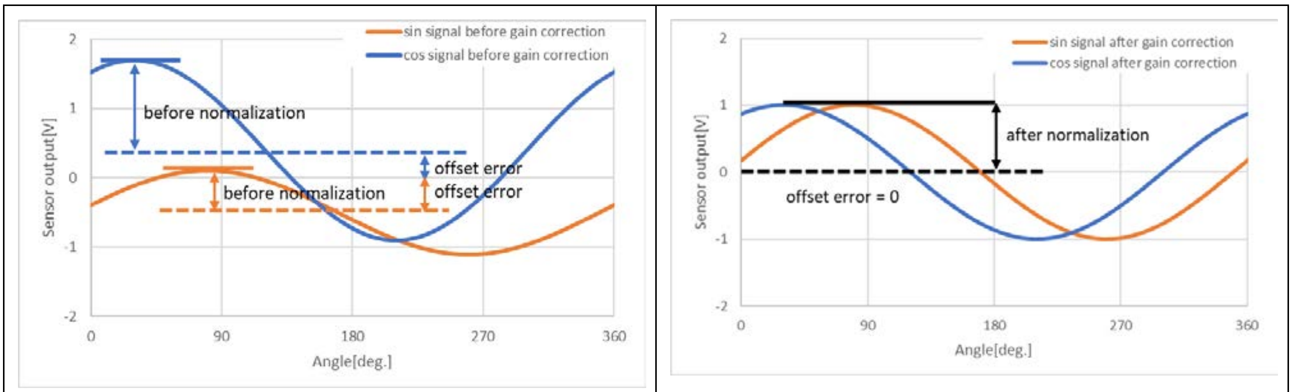


Figure 5-17 Startup Sequence Based on Vector Control by Using the Encoder (Example)

5.2.12 Error Correction of an Analog Magnet Sensor

This control program provides functionality to correct analog outputs of a magnet sensor. When an analog-output sensor is used to detect an angle from sine and cosine signals, an angle error is generated due to the offset or gain of the sensor's outputs and variation in phases. You can use a function of this control program to correct outputs of the sensor. Figure 5-18 shows the concept of gain correction and Figure 5-19 shows the concept of phase correction.



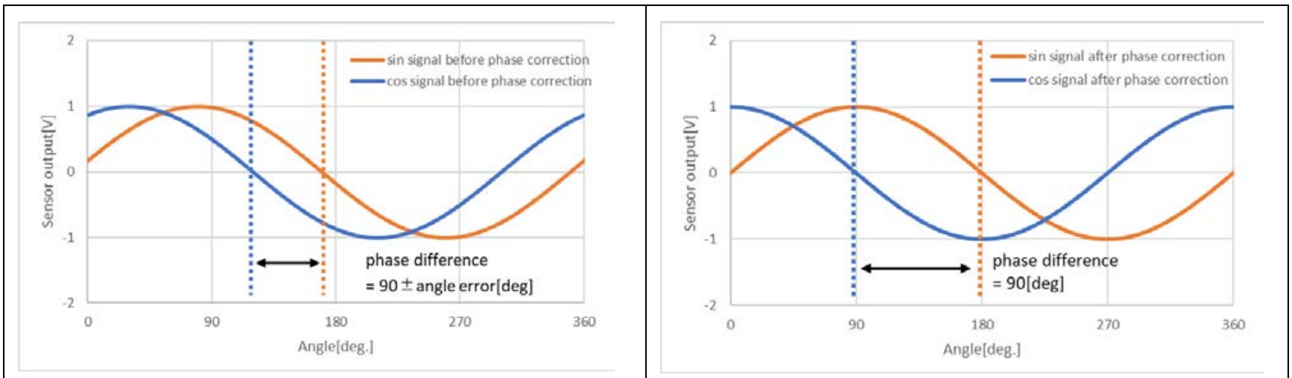
(a) Conceptual image of detection signals before gain correction

(b) Conceptual image of detection signals after gain correction

Figure 5-18 Concept of Gain Correction of Analog Outputs

When the sensor output correction function is enabled, the sensor output data are acquired at the first start-up. Based on the acquired data, correction factors are calculated by software and are used to calculate an angle from the sensor's outputs. Data output from the sensor are acquired in open-loop operation. Figure 5-20 shows the flow of data acquisition and correction.

Regardless of whether the correction function is enabled or disabled, the initial position information is acquired at startup because it is necessary to acquire the position offset between the rotor of the motor and detection magnet of the magnet sensor.



(a) Conceptual image of detection signals before phase correction

(b) Conceptual image of detection signals after phase correction

Figure 5-19 Concept of Phase Correction of Analog Outputs

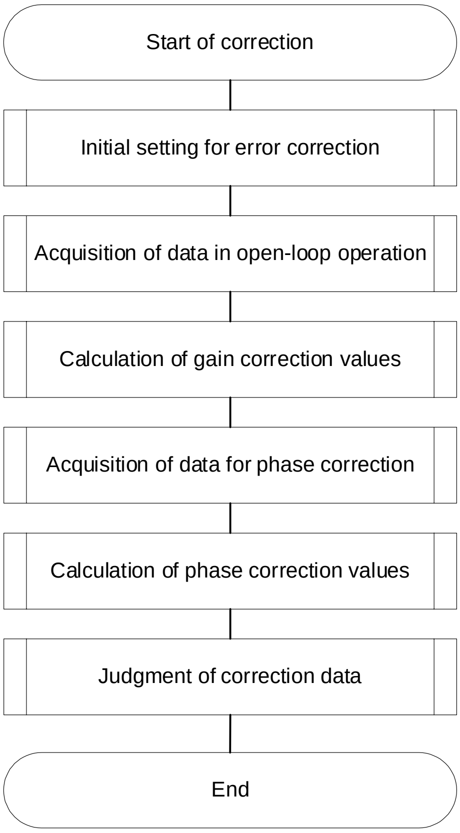


Figure 5-20 Flowchart for Error Correction Processing

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 Functions

Table 5-18 lists the functions of the current control module.

Table 5-18 List of Functions of the Current Control Module

Function	Description
Current control	Makes calculation according to the current command value to set the PWM output value.
Current offset calibration	Calculates the offset value of the current value detected through A/D conversion.
Voltage error compensation	Compensates for the effects of the output voltage dead time.
Coordinate transformation and inverse transformation	Performs coordinate transformation for the current value detected for vector control. This function also performs inverse transformation of the coordinate for the calculation results to restore the original coordinate axis.
Modulation	Improves voltage utilization by modulation to a PWM signal.
Decoupling control	Makes calculation for interference cancellation to prevent interference between the d and q axes.

5.3.2 Module Configuration Diagram

Figure 5-21 shows the module configuration.

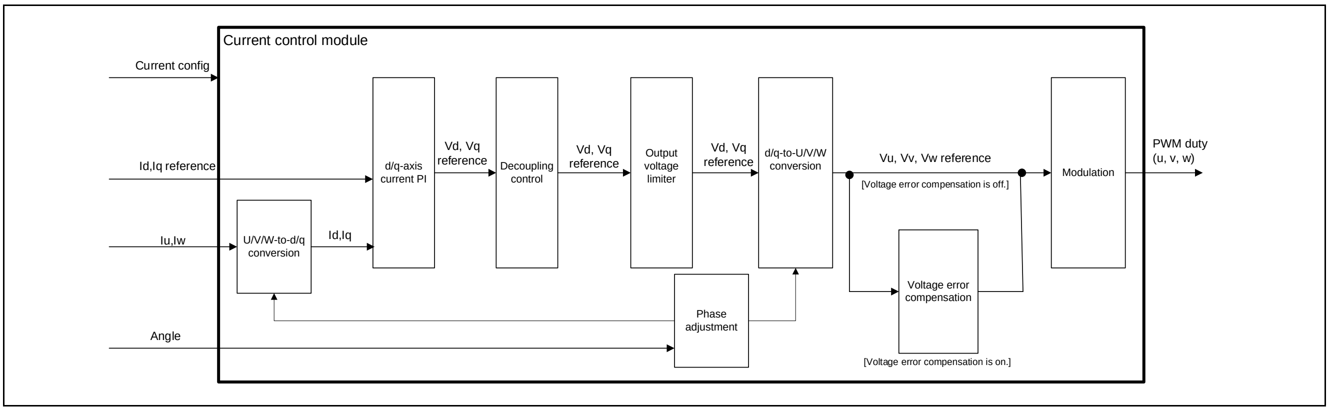


Figure 5-21 Current Control Module Configuration Diagram

5.3.3 Flowchart

Figure 5-22 shows the flowchart for the loop processing of the current control module.

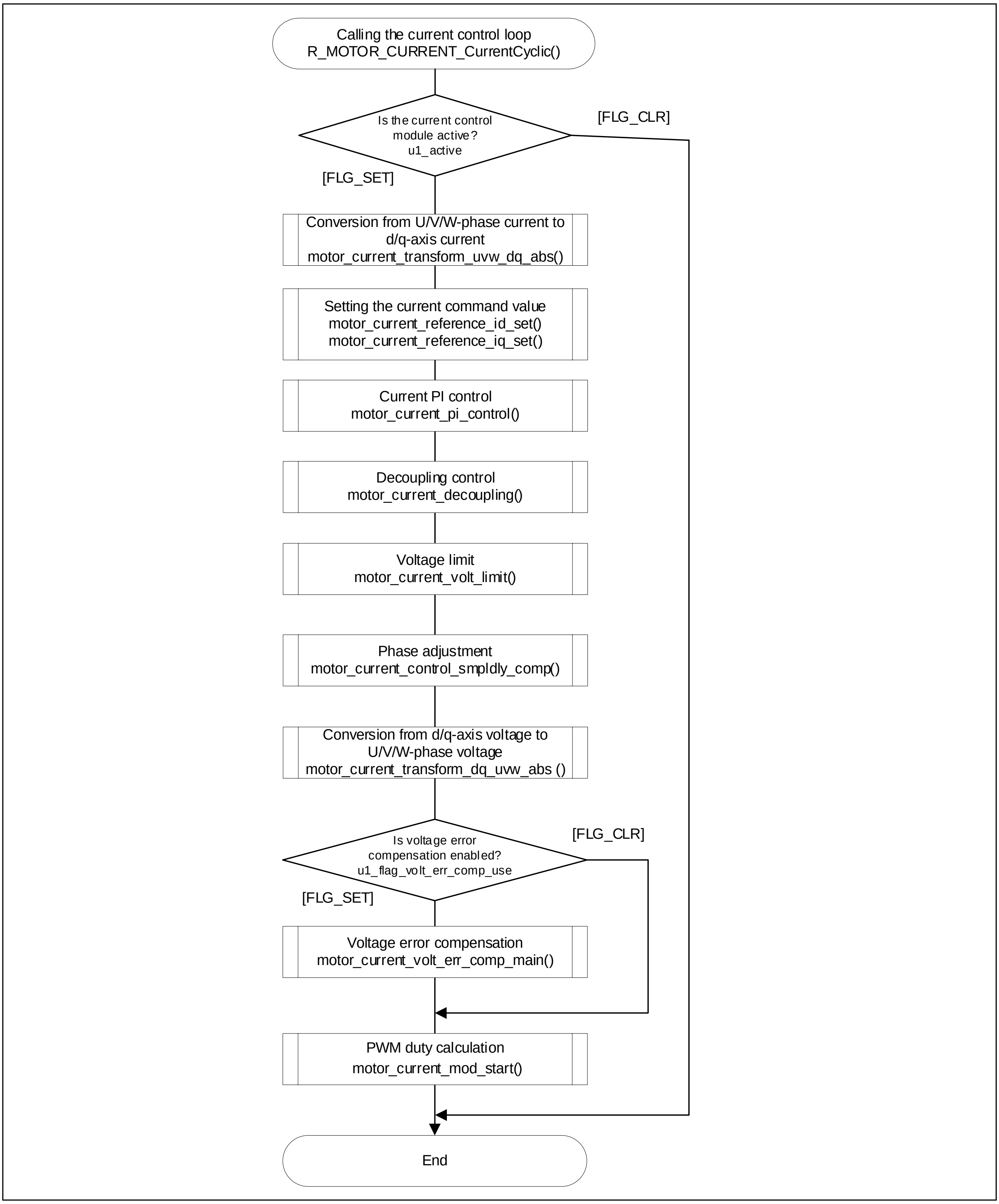


Figure 5-22 Flowchart for the Loop Processing of the Current Control Module

5.3.4 API

Table 5-19 lists the API functions of the current control module.

Table 5-19 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 the 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	Calibrates the offset value for use in current detection.
R_MOTOR_CURRENT_CurrentOffsetRemove	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 5-20 shows the configurations for use with the current control module. Set up the functions to be used and the necessary parameters. Table 5-21 shows the initial values for each microcontroller.

Table 5-20 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. Enabling: MTR_ENABLE Disabling: 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 setting
	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 the current control system [Hz]
	CURRENT_CFG_ZETA	Damping ratio for the current control system
	CURRENT_CFG_REF_ID_OPENLOOP	d-axis current command value in open loop mode [A]
	CURRENT_CFG_ID_UP_STEP_TIME	Setting of the additional time for the d-axis current command value

Table 5-21 List of Initial Values for Configurations

Macro Name	Set Value
CURRENT_CFG_VOLT_ERR_COMP	MTR_ENABLE
CURRENT_CFG_MODULATION_METHOD	MOD_METHOD_SVPWM
CURRENT_CFG_OFFSET_CALC_TIME	RX72M/RX72T/RX66T: 512.0f RX24T/RX24U/RX23T/RX13T: 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	RX72M/RX72T/RX66T: 2560.0f RX24T/RX24U/RX23T/RX13T: 1280.0f

5.3.6 Structure and Variable Information

Table 5-22 lists the structures and variables for use with 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 5-22 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	State of the d axis at startup
	u1_state_iq_ref	State of the q axis at startup
	u1_flag_offset_calc	Flag for the current offset calculation
	u2_offset_calc_time	Measurement time setting in current offset calibration
	u2_crnt_offset_cnt	Measurement count in current offset calibration
	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
	f4_modv	V-phase duty cycle
	f4_modw	W-phase duty cycle

Structure	**Variable**	**Description**
st_current_control_t Structure for the current control module	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_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 value of 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 setting
	f4_ctrl_period	Control interval [s]
	f4_current_omega_hz	Natural frequency for the current control system [Hz]
	f4_current_zeta	Damping ratio for the current control system
	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 output from the current control module	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]
st_current_input_t Structure for input to the current control module	u1_state_id_ref	State of the d axis
	u1_state_iq_ref	State 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]

Structure	**Variable**	**Description**
st_current_input_t Structure for input to the current control module	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 5-23 lists the macros for use with the current control module.

Table 5-23 List of Macros

File Name	Macro Name	Defined Value	Description
r_motor_current_api.h	CURRENT_STATE_ID_ZERO_CONST	0	Current state for the d axis: d-axis current always-0 mode
	CURRENT_STATE_ID_INPUT	1	Current state for the d axis: d-axis current command input mode
	CURRENT_STATE_ID_MANUAL	2	Current state for the d axis: d-axis fixed-command mode
	CURRENT_STATE_ID_UP	3	Current state for the d axis: d-axis current increase mode
	CURRENT_STATE_ID_DOWN	4	Current state for the d axis: d-axis current decrease mode
	CURRENT_STATE_IQ_ZERO_CONST	0	Current state for the q axis: q-axis current always-0 mode
	CURRENT_STATE_IQ_SPEED_PI_OUTPUT	1	Current state 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 damping ratio for the current control system

In the current control module, the control gain is adjusted by tuning the natural frequency for the current control system and the damping ratio for the current control system. Set the natural frequency for the current control system 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 damping ratio for the current control system, 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 damping ratio for the current control system, use the following variables of the st_current_config_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 damping ratio for the current control system can be adjusted by using the RMW.

To set the natural frequency for the current control system, use f4_current_omega_hz. (See Table 5-22.)

To set the damping ratio for the current control system, use f4_current_zeta. (See Table 5-22.)

(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_config_t structure (the 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)

A modulated voltage can be output to improve the voltage utilization. The modulation operation is set from the API of the current control module.

5.4.1 Functionality

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.

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

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

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

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

5.4.2 Configurations

Table 5-24 lists the configurations for the modulation function.

Table 5-24 List of Configurations

File Name	Macro Name	Set Value	Description
r_motor_module_cfg.h	CURRENT_CFG_MODULATION_METHOD	MOD_METHOD_SVPWM	Pulse-width modulation drive mode

5.4.3 Structures

Table 5-25 lists the structures for use with the modulation function.

Table 5-25 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
	u1_sat_flag	Saturation flag

5.4.4 Macro Definition

Table 5-26 lists the macros that are used for the modulation function.

Table 5-26 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 compensates the effects of the output voltage dead time. It operates through the API of the current control module.

5.5.1 Functionality

In the voltage PWM converter, to prevent the switching elements of the upper and lower arms 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.

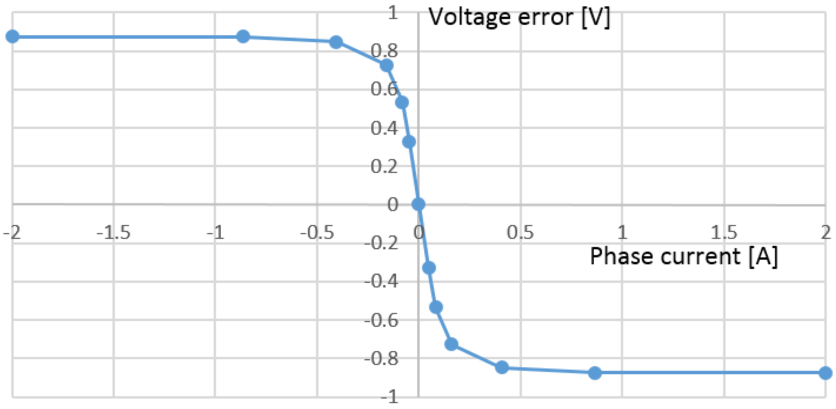


Figure 5-23 Current Dependency in the Voltage Error (Example)

5.5.2 Configurations

Table 5-27 lists the configurations for the voltage error compensation function.

Table 5-27 List of Configurations

File Name	Macro Name	Set Value	Description
r_motor_inverter_cfg.h	INVERTER_CFG_COMP_V0	0.672f	Voltage compensation table
	INVERTER_CFG_COMP_V1	0.945f	Voltage compensation table
	INVERTER_CFG_COMP_V2	1.054f	Voltage compensation table
	INVERTER_CFG_COMP_V3	1.109f	Voltage compensation table
	INVERTER_CFG_COMP_V4	1.192f	Voltage compensation table
	INVERTER_CFG_COMP_I0	0.013f	Voltage compensation table
	INVERTER_CFG_COMP_I1	0.049f	Voltage compensation table
	INVERTER_CFG_COMP_I2	0.080f	Voltage compensation table
	INVERTER_CFG_COMP_I3	0.184f	Voltage compensation table
	INVERTER_CFG_COMP_I4	0.751f	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 parameters 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 flux weakening and the disturbance torque/speed estimation observer, which are handled by the submodules.

5.6.1 Functionality

Table 5-28 lists the functions of the speed control module.

Table 5-28 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.
Flux weakening control setting	Uses the 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 control module.

5.6.2 Module Configuration Diagram

Figure 5-24 shows the module configuration of the speed control module.

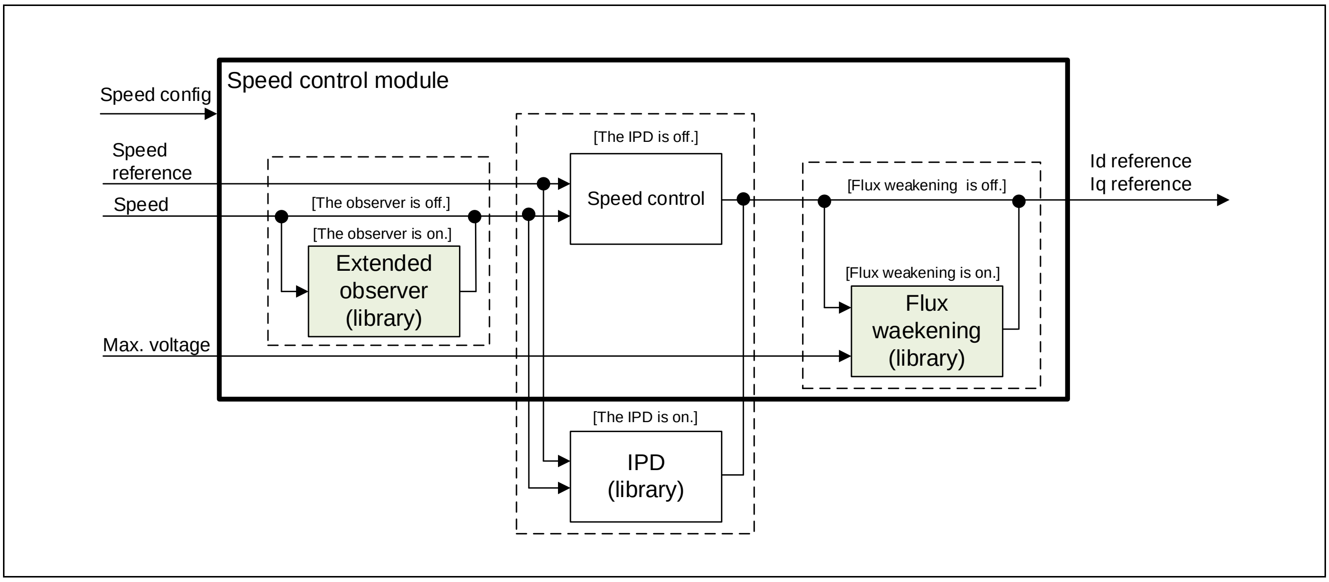


Figure 5-24 Speed Control Module Configuration Diagram

For details about the flux weakening control function and disturbance torque/speed estimation observer function, which are handled by the submodules of the speed control module, see 5.7, Flux Weakening Control (Speed Control Module) and 5.8, Disturbance Torque/Speed Estimation Observer (Speed Control Module).

5.6.3 Flowchart

Figure 5-25 shows the flowchart for speed control.

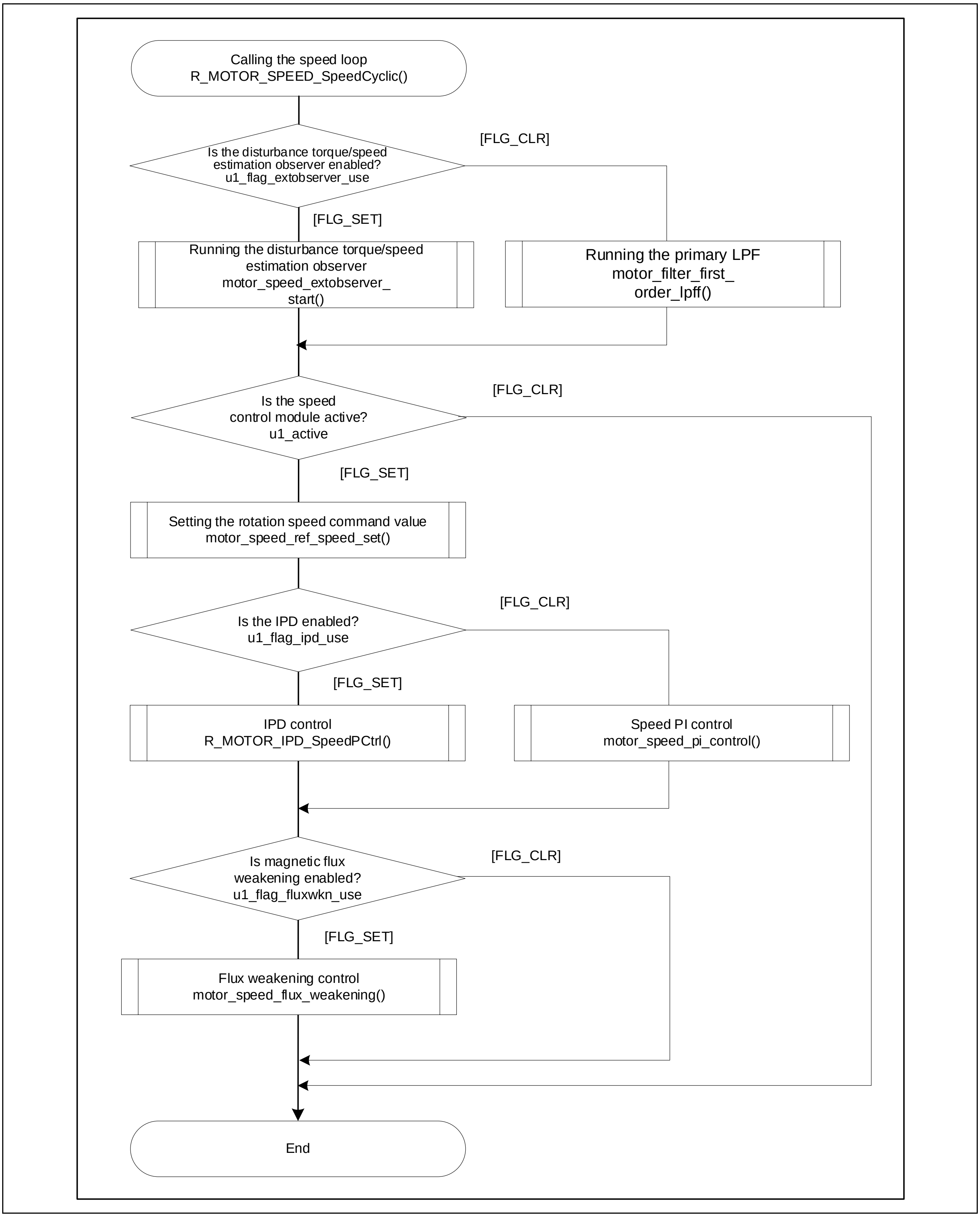


Figure 5-25 Flowchart for Speed Control

5.6.4 API

Table 5-29 lists the API functions of the speed control module.

Table 5-29 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 the 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 control module.
R_MOTOR_SPEED_IPDEnableSet	Enables the IPD control module.
R_MOTOR_SPEED_ExtObserverParameter Update	Updates the control parameters for the disturbance torque/speed estimation observer.

5.6.5 Configurations

Table 5-30 lists the configurations for the speed control module. Set up the functions to be used and the necessary parameters. Table 5-31 shows the initial values for each microcontroller.

Table 5-30 List of Configurations

File Name	Macro Name	Description
r_motor_module_cfg.h	SPEED_CFG_FLUX_WEAKENING	Flux weakening control setting Enabling: MTR_ENABLE Disabling: MTR_DISABLE
	SPEED_CFG_OBSERVER	Disturbance torque/speed estimation observer setting Enabling: MTR_ENABLE Disabling: MTR_DISABLE
	SPEED_CFG_CTRL_PERIOD	Control interval setting [s]
	SPEED_CFG_OMEGA	Natural frequency for the speed control system [Hz]
	SPEED_CFG_ZETA	Damping ratio for the speed control system
	SPEED_CFG_LPF_OMEGA	LPF bandwidth for the speed control system [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 5-31 List of Initial Values for Configurations

Macro Name	RX13T, RX23T, RX24T, RX24U	RX66T, RX72T, RX72M
SPEED_CFG_FLUX_WEAKENING	MTR_DISABLE	
SPEED_CFG_OBSERVER	MTR_DISABLE	
SPEED_CFG_CTRL_PERIOD	0.001f	0.0005f
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 5-32 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 5-32 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 control module
	u1_flag_fluxwkn_use	Flag for whether to use 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_parameter_t	Structure for motor constants
	st_pi_ctrl_t	Structure for PI control
	st_extobs_t	Structure for the disturbance torque/speed estimation observer
	st_fluxwkn_t	Structure for flux weakening control
	st_1st_order_lpf_t	Structure for the LPF
	st_ipd_ctrl_t	Structure for IPD control

Table 5-33 List of Structures and Variables (2)

Structure	Variable	Description
st_speed_config_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 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 the speed control system [Hz]
	f4_speed_zeta	Damping ratio for the speed control system
	f4_speed_lpf_hz	LPF for speed control [Hz]
	st_motor_param_t	Structure for motor constants
st_speed_input_t Structure for input to the speed control module	u1_state_speed_ref	Speed command state
	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 on the d and q axes [V]
st_speed_output_t Structure for output from the speed control module	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 using the 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 5-34 lists the macros of the speed control module.

Table 5-34 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 damping ratio for the speed control system

In the speed control module, the control gain is adjusted by tuning the natural frequency for the speed control system and the damping ratio for the speed control system. Increasing the natural frequency for the speed control system improves the responsiveness, expanding the following capability of the speed to the commanded speed. The maximum settable natural frequency for the speed control system is limited to 1/3 of the maximum settable natural frequency for the current control system to prevent interference with current control. For the damping ratio for the speed control system, 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 damping ratio for the speed control system, use the following variables of the st_speed_config_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 the speed control system, use f4_speed_omega_hz. (See Table 5-33.)
- To set the damping ratio for the speed control system, use f4_speed_zeta. (See Table 5-33.)

(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_config_t structure (the 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 Flux Weakening Control (Speed Control Module)

The 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, when 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. Flux weakening control is a technology that solves this problem.

5.7.1 Functionality

In 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 5-26.

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

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

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

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

I_a: Current vector magnification [A]

Figure 5-26 Formula for Calculating the d-Axis Command Value in 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 (the API function for updating the control parameters for the speed control module) to set "u1_flag_fluxwkn_use" (flag for whether to use 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 expanding the following capability to speed commands and reducing speed ripple.

5.8.1 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 ($\widehat{\omega}$) 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.

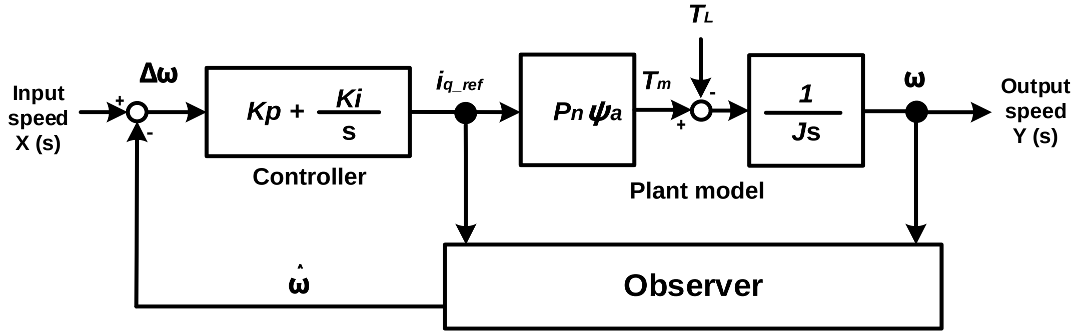


Figure 5-27 Model of the Speed Control System

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 (the 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 natural frequency for the disturbance observer becomes about four to six times the natural frequency for the speed control system.

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. Feed-forward control for the speed command is also implemented to improve the responsiveness of speed control.

5.9.1 Functionality

Table 5-35 lists the functions of the position control module.

Table 5-35 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 feed-forward control	Performs feed-forward (FF) control to the speed command.
Position profiling	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 5-28 shows the module configuration of the position control module.

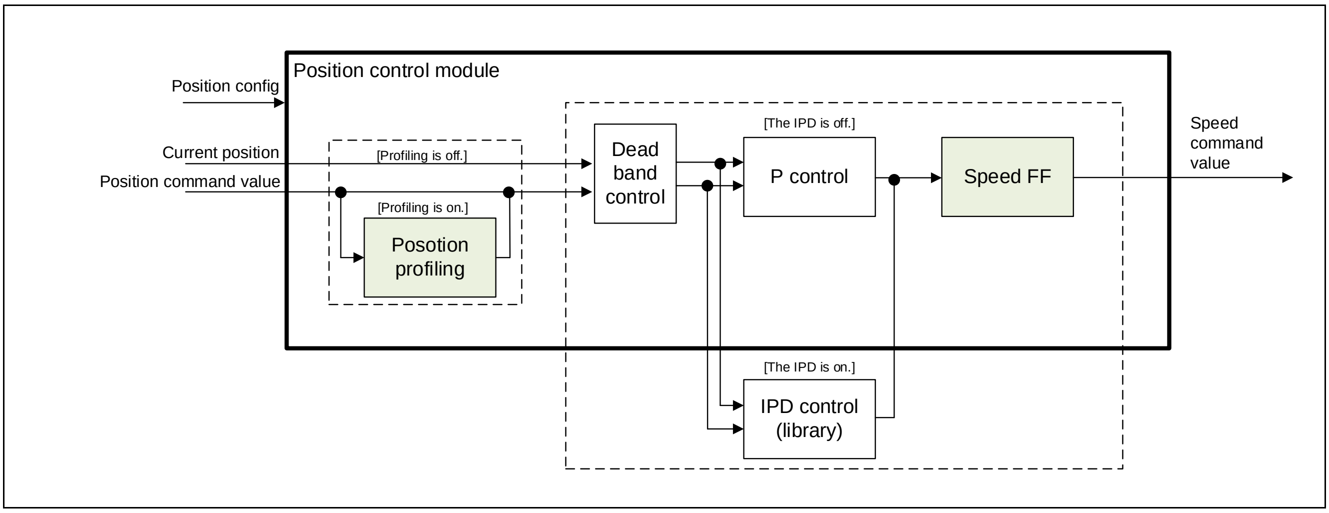


Figure 5-28 Position Control Module Configuration Diagram

5.9.3 Flowcharts

Figure 5-29 shows the flowchart for position control.

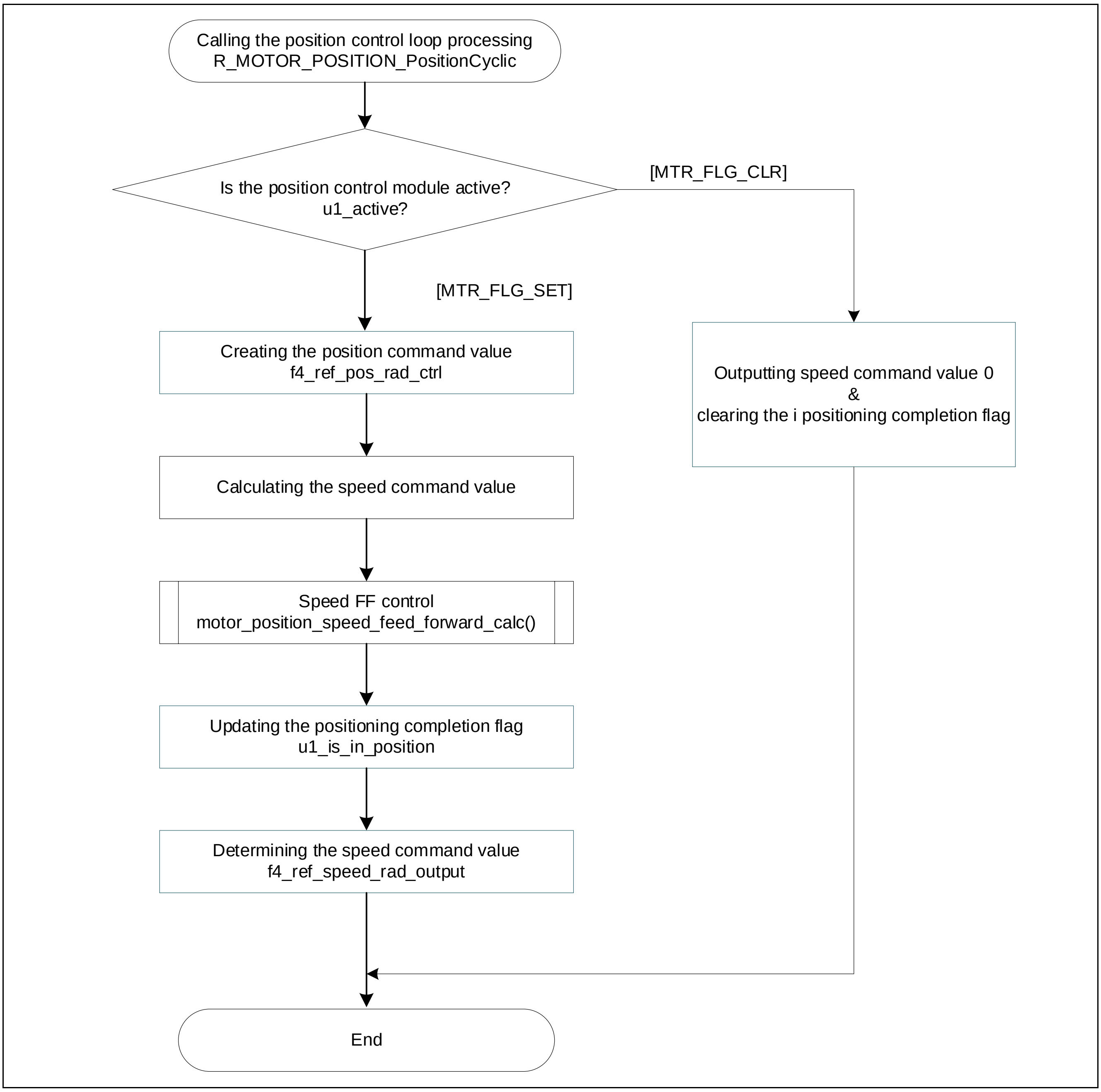


Figure 5-29 Flowchart for the Position Control Loop Processing

For a detailed flow for the position command creation processing, see Figure 5-30.

For a detailed flow for the speed command value calculation processing, see Figure 5-31.

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

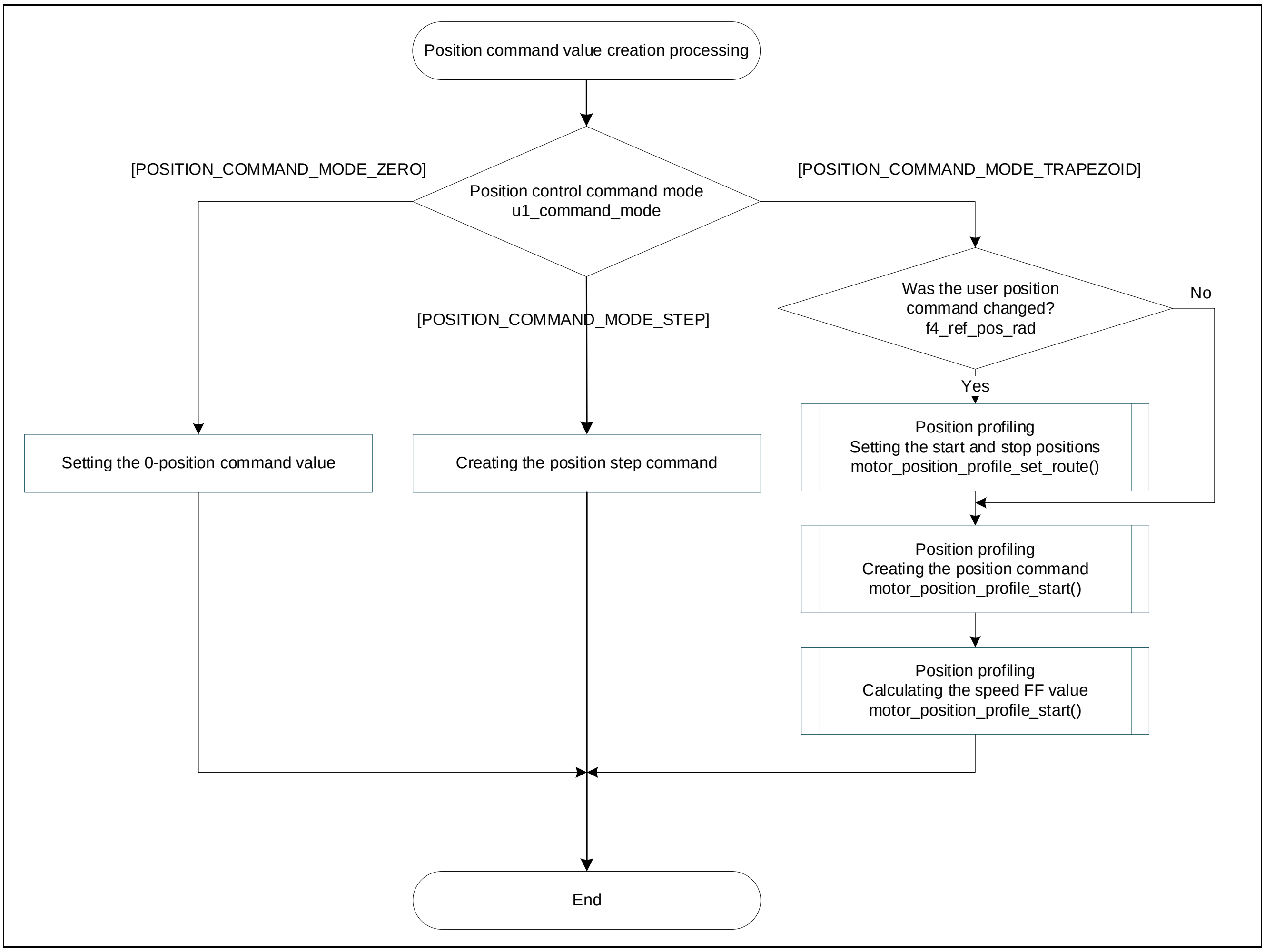


Figure 5-30 Flowchart for the Position Command Creation Processing

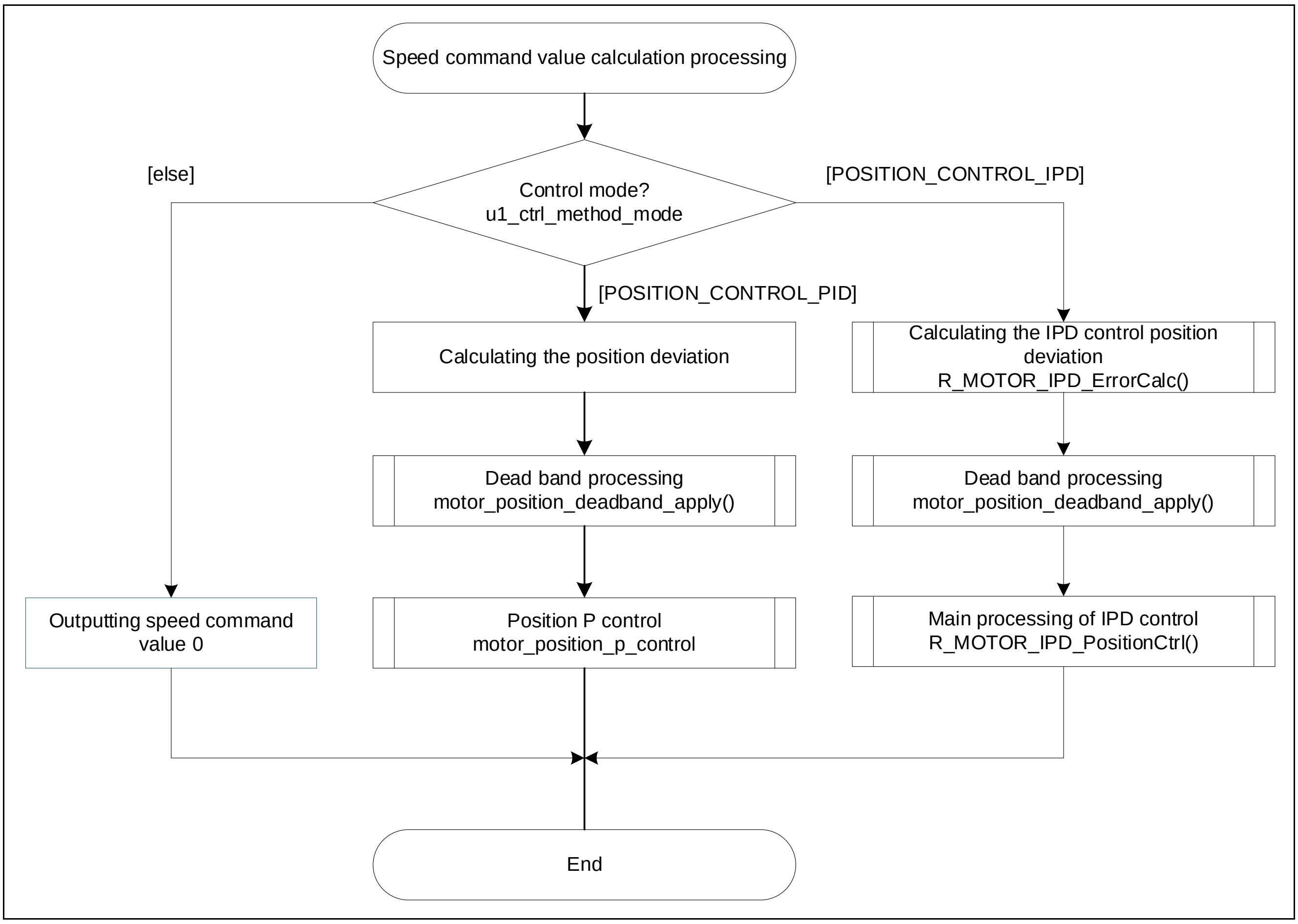


Figure 5-31 Speed Command Value Calculation Processing

5.9.4 Mode Management

(a) Position command modes

Table 5-36 lists the position control command modes.

Table 5-36 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 (the API function for setting the position control command mode).

(b) Control modes

Table 5-37 lists the control modes.

Table 5-37 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 (the API function for enabling the IPD control module).

5.9.5 API Functions

Table 5-38 lists the API functions for the position control module.

Table 5-38 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 the 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 control module.
R_MOTOR_POSITION_IPDEnableSet	Enables a call to the IPD control 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 5-39 shows the configurations for use with the position control module. Set up the functions to be used and the necessary parameters. Table 5-40 shows the initial values for each microcontroller.

Table 5-39 List of Configurations

File Name	Macro Name	Description
r_motor_module_cfg.h	POSITION_CFG_CTRL_PERIOD	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 the position control system [Hz]
	POSITION_CFG_BAND_LIMIT	Range in which the position error is zero (number of position sensor pulses)

Table 5-40 List of Initial Values for Configurations

Macro Name	Set Value
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	RX72M/RX72T/RX66T: 800.0f RX24T/RX24U/RX23T/RX13T: 400.0f
POSITION_CFG_OMEGA	4.0f
POSITION_CFG_BAND_LIMIT	3.0f

5.9.7 Structure and Variable Information

Table 5-41 lists the structures for use with 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 5-41 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 position profiling [rad]
	f4_speed_ff_rad	Speed feed-forward 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 position profiling
	st_motor	Structure for motor constants
	p_st_ipd	IPD control 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
	st_pi_position	Structure for PI control
	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	Stabilization wait time (number of control intervals)
	f4_feedforward_ratio	Speed feed-forward proportionality coefficient
	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]

Structure	Variable	Description
st_position_input_t Structure for input to the position control module	f4_position_rad	Current position [rad]
st_position_output_t Structure for output from the position control module	f4_speed_ref	Speed command output value [rad/s]
	f4_position_err	Position deviation value [rad] Use this variable when you want to check the deviation externally in cases such as when automatic adjustment is used.
	u1_in_position	Positioning completion flag

5.9.8 Macro Definition

Table 5-42 lists the macros for use with the position control module.

Table 5-42 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 the position control system

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

When you set or update the values of the natural frequencies for control systems, use the following variable of the st_position_config_t structure (the structure for setting the parameters for controlling the position control module). After you have set the desired value in this variable, apply it by using R_MOTOR_POSITION_ParameterUpdate (the API function for updating the parameters that control the position control module).

- To set the natural frequency for the position control system, use f4_posprof_max_speed_rad. (See Table 5-32.)

(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_config_t structure (the structure for setting the parameters for controlling the position control module).

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

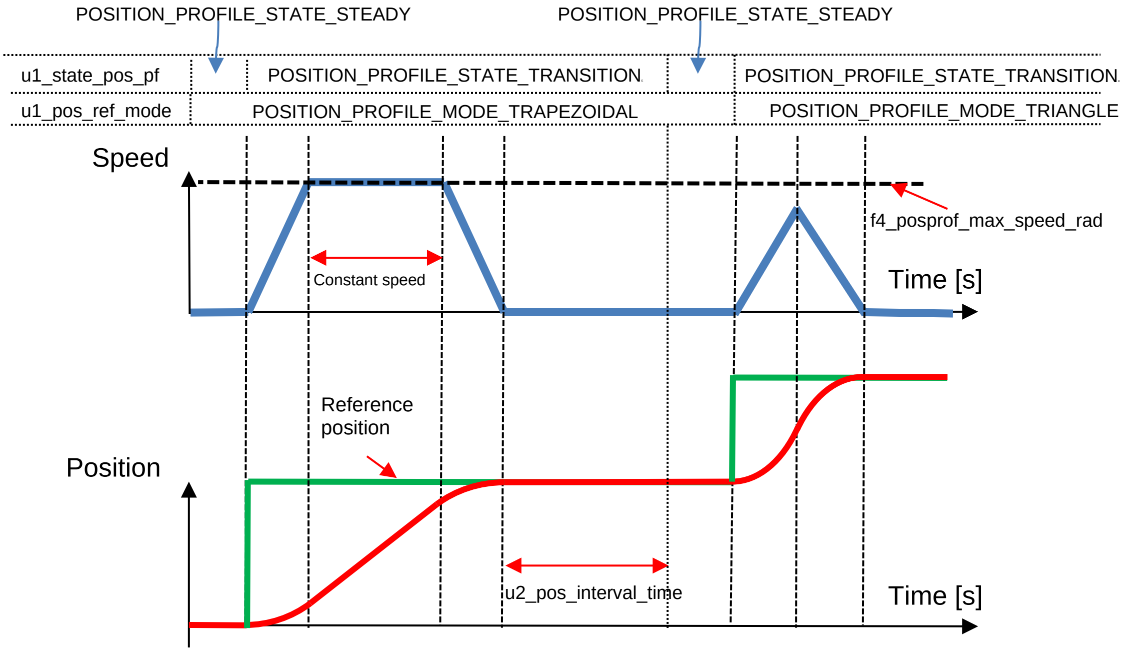
The configurations of the position control module can be specified by using r_motor_module_cfg.h. The values set in this file are applied as initial values at system startup. For details about the items to be set, see 5.9.6, Configurations.

5.10 Position Profiling (Position Control Module)

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

5.10.1 Functionality

The position control module has a function (moving-average acceleration/deceleration algorithm) that controls the speed command value by recalculating the position command value at each control interval based on the position command value that is preset, the acceleration/deceleration time, and the maximum speed. Figure 5-32 shows an overview of this algorithm. When the speed obtained from the position deviation and acceleration time exceeds the maximum speed at the rate of acceleration, a position command value is created so that the speed command values take the trapezoidal form. For the variables and other items shown in Figure 5-32, see Table 5-12 and Table 5-13.



POSITION_PROFILE_MODE_TRIANGLE:
f4_accel_max_speed * f4_accel_time >= |f4_pos_dt_rad|

POSITION_PROFILE_MODE_TRAPEZOIDAL:
f4_accel_max_speed * f4_accel_time < |f4_pos_dt_rad|

Figure 5-32 Shaping of the Position Command Values and Triangular/Trapezoidal Speed Command Values

5.10.2 Configurations

Table 5-43 shows the configurations that are used for the position profile function. Set up the functions to be used and the necessary parameters. Table 5-44 shows the initial values for each microcontroller.

Table 5-43 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 5-44 List of Initial Values for Configurations

Macro Name	Set Value
POSITION_CFG_CTRL_PERIOD	SPEED_CFG_CTRL_PERIOD
POSITION_CFG_INTERVAL_TIME	RX72M/RX72T/RX66T: 800.0f RX24T/RX24U/RX23T/RX13T: 400.0f
MOTOR_CFG_MAX_SPEED_RPM	4000.0f

5.10.3 Structures

Table 5-45 lists the structures for use with 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 5-32. "st_ppf" in this table is the relevant variable.

Table 5-45 List of Variables

Structure	Variable	Description
st_position_profiling_t Structure for position profiling	u1_state_pos_pf	Position profiling state
	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_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 use in profiling
	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 5-46 lists the macros that are used for the position profile function.

Table 5-46 List of Macros

File Name	Macro Name	Defined Value	Remarks
r_motor_position_profiling.h	POS_PROFILE_ACCEL_TIME	0.3f	Acceleration time for the speed command value [s]
	POS_PROFILE_CTRL_TRIANGLE	0	Triangular wave control mode
	POS_PROFILE_CTRL_TRAPEZOIDAL	1	Trapezoidal wave control mode
	POS_PROFILE_STEADY_STATE	0	Steady state
	POS_PROFILE_TRANSITION_STATE	1	Transition state

5.10.5 Adjustment and Configuration of Parameters

(a) Setting the parameters for control

A position command value considering the acceleration/deceleration time 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 the variables are used as the parameters that adjust the operation shown in Figure 5-32.

- Acceleration time: f4_accel_time
- Maximum speed: f4_posprof_max_speed_rad
- Stabilization wait time: u2_pos_interval_time

5.11 IPD Control Module

5.11.1 Functionality

In the position control system, 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. Reducing 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 differentiation act on only the operation amount (controller output). With this method, increased responsiveness is possible while mitigating vibration that occurs during positioning.

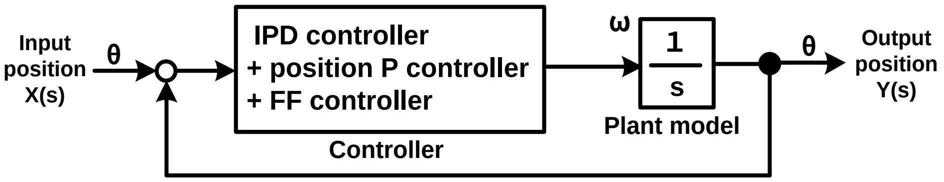


Figure 5-33 Model of IPD Control (Position Control)

The IPD control that is actually implemented is combined with normal proportion control and feed-forward control.

5.11.2 API

Table 5-47 lists the API functions for the IPD control module.

Table 5-47 List of API Functions

API Function	Description
R_MOTOR_IPD_Open	Generates an instance of the IPD control module.
R_MOTOR_IPD_Close	Places the module in the 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 control 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 control module.
R_MOTOR_IPD_PositionSync	Updates the internal position variable information for use in IPD control.

5.11.3 Structure and Variable Information

Table 5-48 lists the structures and variables for the IPD control module. In the IPD control module, the structure for the IPD control module (g_st_ipd) is defined by securing an instance of the module from the API.

Table 5-48 List of Structures and Variables

Structure	Variable	Description
st_ipd_ctrl_t Structure for the IPD control module	u1_ipd_lpf_flag	Flag for whether the 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 feedback (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 damping ratio
	st_ipd_2nd_lpf	Secondary LPF structure

5.11.4 Macro Definition

Table 5-49 lists the macros for the IPD control module.

Table 5-49 List of Macros

File Name	Macro Name	Defined Value	Remarks
r_motor_ipd_api.h	IPD_LPF_OMEGA	200.0f	Natural frequency for the position LPF [Hz]
	IPD_LPF_ZETA	1.0f	Damping ratio for the position LPF
	IPD_SPEED_RATIO	2.5f	Speed gain coefficient
	IPD_POS_FF_RATIO	0.9f	Position feed-forward 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 the 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 5-50 shows the values set in the sample program.

Table 5-50 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 (Magnet Sensor)

The sensor module calculates the position and speed of the motor. In the sample program, the sensor module is configured to support analog output, digital output, and SPI output from a magnet sensor. The sensor module receives each of the signals from the magnet sensor, calculates the position and speed, and then outputs the results. The sensor output setting can be switched by modifying the relevant configuration. When an inductive sensor is used, use the configuration for the analog magnet sensor output. This is because the position and speed are calculated in the same way with the analog output magnet sensor and inductive sensor.

5.12.1 Functions

Table 5-51 lists the functions of the sensor module.

Table 5-51 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 5-34 shows the sensor module configuration.

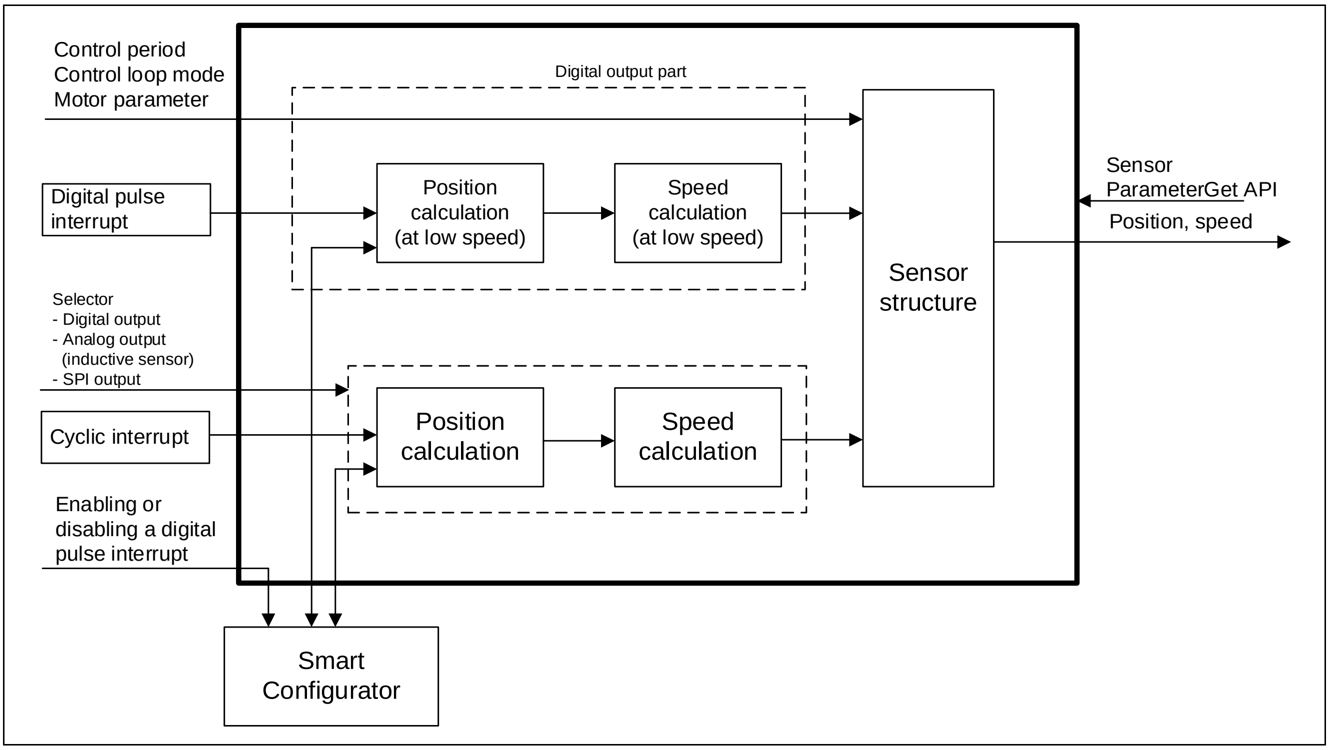


Figure 5-34 Sensor Module Configuration Diagram

5.12.3 Flowchart

Figure 5-35 shows the flowchart for calculating the position and speed from input capture interrupts for digital output pulses. In the case of analog output or SPI output, there is no dedicated interrupt processing.

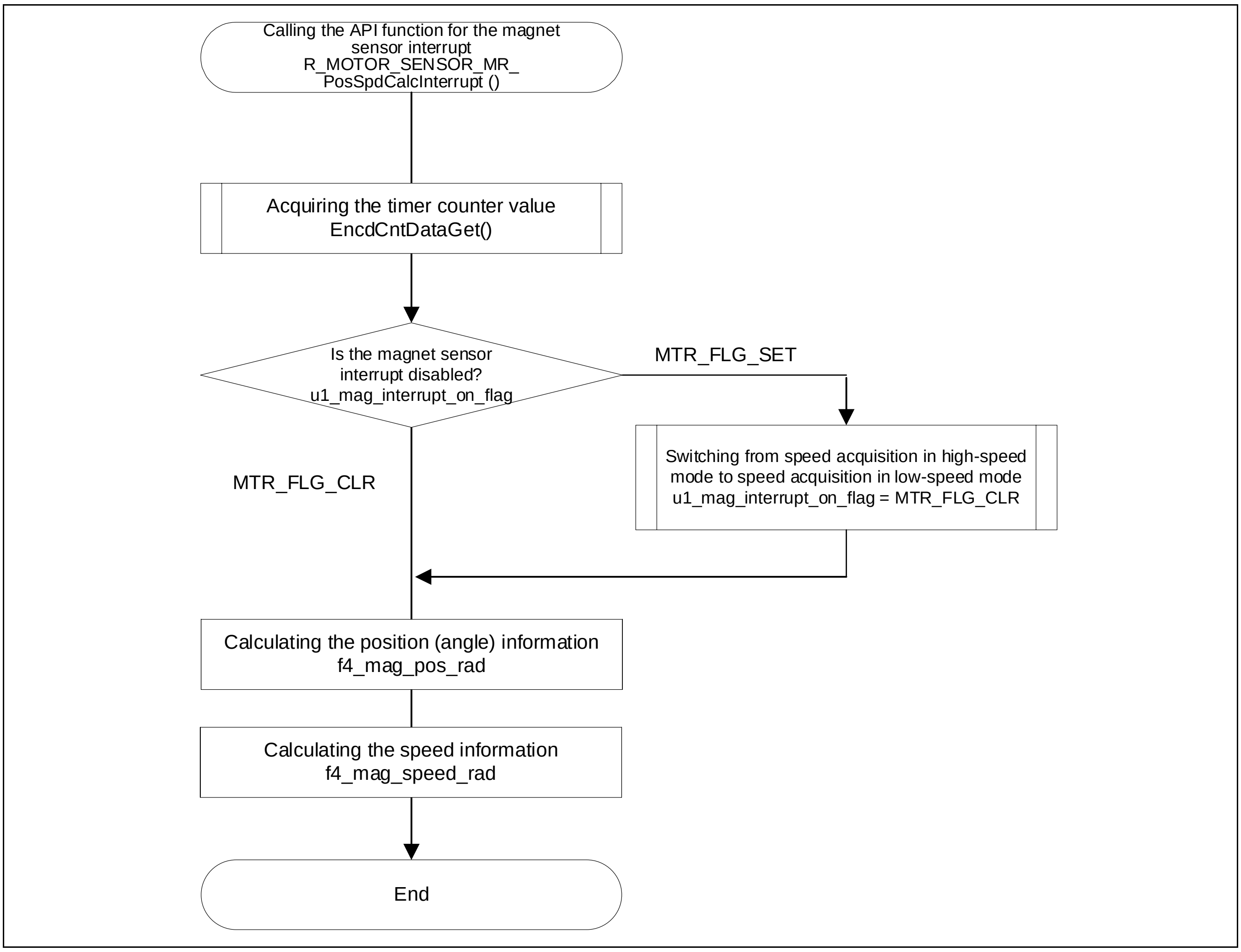


Figure 5-35 Interrupt Processing Flowchart for Digital Output Pulse Signals

5.12.4 API

Table 5-52 lists the API functions for the sensor module.

Table 5-52 List of API Functions

API Function	Description
R_MOTOR_SENSOR_MR_Open	Generates an instance of the sensor module. Run this API function first when using this module.
R_MOTOR_SENSOR_MR_Close	Places the sensor module in the reset state.
R_MOTOR_SENSOR_MR_Reset	Initializes the module.
R_MOTOR_SENSOR_MR_ParameterGet	Acquires position and speed information from the sensor.
R_MOTOR_SENSOR_MR_ParameterUpdate	Updates the control parameters of the sensor module.
R_MOTOR_SENSOR_MR_DriverParameterUpdate	Sets the corresponding Smart Configurator functions in the sensor module.
R_MOTOR_SENSOR_MR_PosSpdCalcInterrupt	Calculates the position and speed from the interval of signal interrupts for digital output.
R_MOTOR_SENSOR_MR_PosSpdCalc	Calculates the position and speed from the angle of the sensor at a fixed interval.
R_MOTOR_SENSOR_MR_HsCountDiffCalc	Acquires the differential of the value of the timer that counts the number of pulses for digital output.
R_MOTOR_SENSOR_MR_PosAdjustInit	Sets the initial position of the sensor output.
R_MOTOR_SENSOR_MR_SpiDtcBufferSet	Sets the DTC buffer for SPI output.
R_MOTOR_SENSOR_MR_AnalogAdcGet	Acquires the sensor output value for analog output and the inductive sensor.

5.12.5 Configurations

Table 5-53 lists the configurations for the sensor module. Set up the functions to be used and the necessary parameters. Table 5-54 shows the initial values for each microcontroller.

Table 5-53 List of Configurations

File Name	Macro Name	Description
r_motor_module_cfg.h	SENSOR_CFG_MR_CONTROL	Sets the output type of the magnet sensor to be used.
	SENSOR_CFG_MR_DIGITAL_TIMER_FREQ	Sets the frequency of the timer for capturing signals for digital output.
	SENSOR_CFG_ MR_DIGITAL_TIMER_BIT	Sets the number of bits in the timer to be used to support digital output.
	SENSOR_CFG_MR_DIGITAL_PPR	Sets the number of pulses [p/r] of the timer to be used to support digital output.
	SENSOR_CFG_MR_DIGITAL_RESOLUTION_MULTIPL	Sets the multiplication factor of digital output signals.
	SENSOR_CFG_MR_POLE_PAIRS	Sets the number of sensor pole pairs.
	SENSOR_CFG_MR_DIGITAL_FUNC_CNT_GET	Sets the function that acquires the timer counter value at the time of input capturing of the digital output signal.
	SENSOR_CFG_MR_DIGITAL_FUNC_CNT_SET	Sets the function that sets the timer counter value at the time of input capturing of the digital output signal.
	SENSOR_CFG_MR_DIGITAL_FUNC_INT_ENABLE	Sets the function that enables input capture interrupts of the digital output signals.
	SENSOR_CFG_MR_DIGITAL_FUNC_INT_DISABLE	Sets the function that disables input capture interrupts of the digital output signals.
	SENSOR_CFG_MR_DIGITAL_FUNC_SPD_TIMER_START	Sets the function that starts the counting of the free-run timer for speed calculation.
	SENSOR_CFG_MR_DIGITAL_FUNC_SPD_TIMER_CNT_GET	Sets the function that acquires the counter value of the free-run timer for speed calculation.
	SENSOR_CFG_MR_ANALOG_CALIB	Sets whether or not to perform automatic adjustment of analog output signals.
	SENSOR_CFG_MR_ANALOG_FUNC_ADC_GET	Sets the function that acquires analog output signals after A/D conversion.
	SENSOR_CFG_MR_SPI_FUNC_DTC_BUFFER_SET	Sets the function that sets the DTC buffer to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_ANGLE_GET	Sets the function that acquires the angle to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_TRANSMIT_BUFFER_SET	Sets the function that sets the 16-bit transmit buffer to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_TRANSMIT_BUFFER2_SET	Sets the function that sets the 32-bit transmit buffer to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_RECEIVE_BUFFER_GET	Sets the function that sets the 16-bit receive buffer to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_RECEIVE_BUFFER2_GET	Sets the function that sets the 32-bit receive buffer to be used to support SPI output.
	SENSOR_CFG_MR_SPI_FUNC_STREAM_COMMAND	Sets the function that sets the streaming mode to be used to support SPI output.

Table 5-54 List of Initial Values for Configurations

Macro Name	RX13T	RX23T	RX24T, RX24U	RX66T	RX72T	RX72M
SENSOR_CFG_MR_CONTROL	SENSOR_CFG_MR_CONTROL_DIGITAL					
SENSOR_CFG_MR_DIGITAL_TIMER_FREQ	20			160	200	120
SENSOR_CFG_MR_DIGITAL_TIMER_BIT	16			32		
SENSOR_CFG_MR_DIGITAL_PPR	1000					
SENSOR_CFG_MR_DIGITAL_RESOLUTION_MULTIPL	4					
SENSOR_CFG_MR_POLE_PAIRS	1					
SENSOR_CFG_MR_DIGITAL_FUNC_CNT_GET	R_Config_MTU1_TcntGet					
SENSOR_CFG_MR_DIGITAL_FUNC_CNT_SET	R_Config_MTU1_TcntSet					
SENSOR_CFG_MR_DIGITAL_FUNC_INT_ENABLE	R_Config_MTU0_InterruptEnable					
SENSOR_CFG_MR_DIGITAL_FUNC_INT_DISABLE	R_Config_MTU0_InterruptDisable					
SENSOR_CFG_MR_DIGITAL_FUNC_SPD_TIMER_START	MTU0*1			GPT3*1	GPT3*1	GPT3*1
SENSOR_CFG_MR_DIGITAL_FUNC_SPD_TIMER_CNT_GET	MTU0*2			GPT3*2	GPT3*2	GPT3*2
SENSOR_CFG_MR_ANALOG_CALIB	MTR_DISABLE					
SENSOR_CFG_MR_ANALOG_FUNC_ADC_GET	Not supported		S12AD0*3	S12AD1*3		
SENSOR_CFG_MR_SPI_FUNC_DTC_BUFFER_SET	Not supported	RSPI0*4				RSPI2*4
SENSOR_CFG_MR_SPI_FUNC_ANGLE_GET	Not supported	RSPI0*5				RSPI2*5
SENSOR_CFG_MR_SPI_FUNC_TRANSMIT_BUFFER_SET	Not supported	RSPI0*6				RSPI2*6
SENSOR_CFG_MR_SPI_FUNC_TRANSMIT_BUFFER2_SET	Not supported	RSPI0*7				RSPI2*7
SENSOR_CFG_MR_SPI_FUNC_RECEIVE_BUFFER_GET	Not supported	RSPI0*8				RSPI2*8
SENSOR_CFG_MR_SPI_FUNC_RECEIVE_BUFFER2_GET	Not supported	RSPI0*9				RSPI2*9
SENSOR_CFG_MR_SPI_FUNC_STREAM_COMMAND	Not supported	RSPI0*10				RSPI2*10

In Table 5-54, each peripheral name indicated with a footnote number replaces the xxxx portion of the following function names:

Notes: 1. R_Config_xxxx_SpeedCalcTimerStart
2. R_Config_xxxx_TcntGet
3. R_Config_xxxx_AdcGet
4. R_Config_xxxx_DtcBufferSet
5. R_Config_xxxx_AngleGet
6. R_Config_xxxx_TxBufferSet
7. R_Config_xxxx_TxBuffer2Set

8. R_Config_xxxx_RxBufferGet
9. R_Config_xxxx_RxBuffer2Get
10. R_Config_xxxx_StreamCommand

5.12.6 Structure and Variable Information

Table 5-55 lists the structures and variables for the sensor module.

Table 5-55 List of Structures and Variables

Structure	Variable	Description
st_sensor_t Structure for the sensor module	-	The variables selected for the structures to be used to support digital output, structures to be used to support analog output, and structures to be used to support SPI output shall apply.
st_sensor_digital_t Structure for digital output	u1_ctrl_loop_mode	Control mode selection
	f4_ctrl_period	Control interval information [s]
	st_dc	Structure for digital output during low-speed rotation
	st_dhc	Structure for digital output during high-speed rotation
	st_motor	Structure for the motor parameters
	*DigitalCntDataGet	Pointer to the function that acquires the count value of the timer that uses the input capture to acquire digital output pulse signals
	*DigitalCntDataSet	Pointer to the function that sets the count value of the timer that uses the input capture to acquire digital output pulse signals
	*DigitalInterruptEnable	Pointer to the function that enables input capture interrupts of the digital output pulse signals
	*DigitalInterruptDisable	Pointer to the function that disables input capture interrupts of the digital output pulse signals
	*DigitalTimerStart	Pointer to the function that starts the counting of the free-run timer for speed calculation
	*DigitalTimerGet	Pointer to the function that acquires the counter value of the free-run timer for speed calculation
st_digital_t Structure for digital output during low-speed rotation	u2_mag_pre_phase_cnt;	Phase counter value of the encoder (used at switchover between low speed and high speed)
	u4_mag_timer_cap_tcnt;	Counter value in input capture of the encoder
	u4_mag_timer_cap_pre_tcnt;	Previous counter value in input capture of the encoder
	u2_mag_timer_cnt_num;	Number of the buffer for speed calculation at low speed
	u2_mag_cpr_mech;	Number of pulses per rotation [p/r]
	u4_mag_timer_cnt_buff;	Buffer for speed calculation at low speed
	u4_mag_pulse_width;	Pulse interval of the encoder for speed detection
	u4_mag_pulse_width_buff;	Buffer for holding the pulse interval of the encoder
	u4_mag_pulse_width_sum;	Total pulse interval of the encoder. This variable is used to detect speed 0.
	s4_mag_angle_cnt;	Counter value of the encoder for the position. This variable is used to integrate the differentials of counter values at intervals
	f4_mag_angle_diff;	Angle information for one pulse width of the encoder
	f4_mag_cpr_mech_inv;	Inverse number of the number of pulses per motor rotation
	f4_mag_speed_pre_rad;	Previous speed [rad/s]
	f4_mag_speed_rad;	Speed (during low-speed rotation) [rad/s]
	f4_mag_pos_rad;	Position (during low-speed rotation) [rad]

Structure	Variable	Description
st_digital_highspeed_t Structure for digital output during high-speed rotation	u1_mag_pos_speed_calc_mode;	Selection of the position and speed detection method for digital output
	u1_mag_interrupt_on_flag;	Interrupt flag for digital output signals
	u1_mag_pos_speed_calc_cnt;	Counter value for position and speed calculation from digital output signals
	u2_mag_hs_pre_phase_cnt;	Counter value at the previous interval for digital output
	s4_mag_hs_angle_cnt;	Position information counter value for digital output
	f4_mag_hs_pos_rad;	Position (during high-speed rotation) [rad]
	f4_mag_hs_pos_pre_rad;	Rotor position information at the previous interval [rad]
	f4_mag_hs_speed_rad;	Speed (during high-speed rotation) [rad/s]
	f4_mag_hs_speed_pre_rad;	Speed at the previous interval [rad/s]
	f4_mag_hs_sw_speed_rad;	Speed threshold between high-speed and low-speed modes [rad/s]
	f4_mag_hs_sw_speed_margin_rad;	Margin around the threshold between high-speed and low-speed modes [rad/s]
st_sensor_analog_t Structure for analog output	u1_ctrl_loop_mode	Control mode selection
	f4_ctrl_period	Control interval information [s]
	u1_mag_speed_calc_counter	Counter for use in skipping the speed calculation
	s2_mag_cycle_cnt	Counter for counting the number of times the control processing proceeds
	s2_mag_pos_rad_pre_1cyc	Angle information at the previous interval [rad]
	f4_mag_sin_offset	Offset value for the sine signal [V]
	f4_mag_cos_offset	Offset value for the cosine signal [V]
	f4_mag_sin_gain	Correction gain value for the sine signal
	f4_mag_cos_gain	Correction gain value for the cosine signal
	f4_mag_magnet_phase_offset	Position offset for the detection magnet [rad]
	f4_mag_add_cos_sin_gain	Correction gain value for cos + sin
	f4_mag_sub_cos_sin_gain	Correction gain value for cos - sin
	f4_mag_pos_rad	Position of the sensor [rad]
	f4_mag_pos_rad_pre	Previous position value [rad]
	f4_mag_speed_rad	Calculated speed [rad/s]
	f4_mag_speed_rad_pre	Previous speed value [rad/s]
	f4_mag_pos_rad_offset	Position offset value [rad]
	f4_mag_speed_lpf	Speed after using the LPF [rad/s]
	f4_mag_angle_rad	Angle of the sensor [rad]
	st_motor	Structure for motor parameters
	*AnalogAdcGet	Pointer to the function that acquires sensor signals through A/D conversion.

Structure	**Variable**	**Description**
st_sensor_spi_t Structure for SPI output	u1_ctrl_loop_mode	Control mode selection
	f4_ctrl_period	Control interval information [s]
	u1_mag_speed_calc_counter	Counter for use in skipping the speed calculation
	s2_mag_cycle_cnt	Counter for counting the number of times the control processing proceeds
	s2_mag_pos_rad_pre_1cyc	Angle information at the previous interval [rad]
	f4_mag_magnet_phase_offset	Position offset for the detection magnet [rad]
	f4_mag_pos_rad	Position of the sensor [rad]
	f4_mag_pos_rad_pre	Previous position value [rad]
	f4_mag_speed_rad	Calculated speed [rad/s]
	f4_mag_speed_rad_pre	Previous speed value [rad/s]
	f4_mag_pos_rad_offset	Position offset value [rad]
	f4_mag_speed_lpf	Speed after using the LPF [rad/s]
	f4_mag_angle_rad	Angle of the sensor [rad]
	u2_mag_angle_spi	Angle acquired by the SPI [rad]
	u1_mag_status_spi	State of the SPI
	u1_mag_crc_spi	CRC setting for SPI streaming mode
	st_motor	Structure for motor parameters
	*SpiDtcBufferSet	Pointer to the function that sets the DTC buffer to be used to support SPI output
	*SpiAngleGet	Pointer to the function that acquires the angle to be used to support SPI output
	*SpiTransmitBufferSet	Pointer to the function that sets the 16-bit transmit buffer to be used to support SPI output
	*SpiTransmitBuffer2Set	Pointer to the function that sets the 32-bit transmit buffer to be used to support SPI output
	*SpiReceiveBufferGet	Pointer to the function that sets the 16-bit receive buffer to be used to support SPI output
	*SpiReceiveBuffer2Get	Pointer to the function that sets the 32-bit receive buffer to be used to support SPI output
	*SpiStreamCommand	Pointer to the function that sets the streaming mode to be used to support SPI output
st_sensor_mr_cfg_t Structure for setting the parameters for controlling the sensor module	-	The variables selected for the structures for setting the parameters to be used to support digital output, structures for setting the parameters to be used to support analog output, and structures for setting the parameters to be used to support SPI output shall apply.
st_sensor_digital_cfg_t Structure for setting the parameters to be used to support digital output	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_mr_cpr;	Number of pulses per rotation [p/r]
	f4_ctrl_period;	Control interval [s]
	st_motor_parameter_t	Structure for the motor parameters

Structure	Variable	Description
st_sensor_analog_cfg_t Structures for setting the parameters to be used to support analog output	u1_ctrl_loop_mode	Control mode (position and speed)
	f4_ctrl_period	Control interval [s]
	f4_mag_sin_offset	Offset value for the sine signal [V]
	f4_mag_cos_offset	Offset value for the cosine signal [V]
	f4_mag_sin_gain	Correction gain value for the sine signal
	f4_mag_cos_gain	Correction gain value for the cosine signal
	f4_mag_add_cos_sin_gain	Correction gain value for cos + sin
	f4_mag_sub_cos_sin_gain	Correction gain value for cos - sin
	st_motor_parameter_t	Structure for the motor parameters
st_sensor_spi_cfg_t Structures for setting the parameters to be used to support SPI output	u1_ctrl_loop_mode	Control mode (position and speed)
	f4_ctrl_period	Control interval [s]
	st_motor_parameter_t	Structure for the motor parameters
st_sensor_mr_output_t Structure for output from the sensor module	-	The variables of the selected sensor output (digital, analog, or SPI) structures shall apply.
st_sensor_digital_output_t Structure for sensor output (digital)	f4_speed_rad;	Speed [rad/s]
	f4_pos_rad;	Position [rad]
	f4_angle_rad	Angle [rad]
st_sensor_analog_output_t Structure for sensor output (analog)	f4_speed_rad;	Speed [rad/s]
	f4_pos_rad;	Position [rad]
	f4_angle_rad	Angle [rad]
st_sensor_spi_output_t Structure for sensor output (SPI)	f4_speed_rad;	Speed [rad/s]
	f4_pos_rad;	Position [rad]
	f4_angle_rad	Angle [rad]
st_sensor_mr_driver_cfg_t Sensor	-	The variables of the selected sensor driver structures (for digital, analog, or SPI output) shall apply.
st_digital_driver_cfg_t Structure for setting the function pointer for digital output	*DigitalCntDataGet	Pointer to the function that acquires the timer counter value at the time of input capturing of the digital output signal
	*DigitalCntDataSet	Pointer to the function that sets the timer counter value at the time of input capturing of the digital output signal
	*DigitalInterruptEnable	Pointer to the function that enables input capture interrupts of the digital output signals
	*DigitalInterruptDisable	Pointer to the function that disables input capture interrupts of the digital output signals
	*DigitalTimerStart	Pointer to the function that starts the counting of the free-run timer for speed calculation
	*DigitalTimerGet	Pointer to the function that acquires the counter value of the free-run timer for speed calculation

Structure	Variable	Description
st_analog_driver_cfg_t Structure for setting the function pointer for analog output	*AnalogAdcGet	Pointer to the function that acquires sensor signals through A/D conversion.
st_spi_driver_cfg_t Structure for setting the function pointer for SPI output	*SpiDtcBufferSet	Pointer to the function that sets the DTC buffer to be used to support SPI output
	*SpiAngleGet	Pointer to the function that acquires the angle to be used to support SPI output
	*SpiTransmitBufferSet	Pointer to the function that sets the 16-bit transmit buffer to be used to support SPI output
	*SpiTransmitBuffer2Set	Pointer to the function that sets the 32-bit transmit buffer to be used to support SPI output
	*SpiReceiveBufferGet	Pointer to the function that sets the 16-bit receive buffer to be used to support SPI output
	*SpiReceiveBuffer2Get	Pointer to the function that sets the 32-bit receive buffer to be used to support SPI output
	*SpiStreamCommand	Pointer to the function that sets the streaming mode to be used to support SPI output

5.12.7 Macro Definition

Table 5-56 lists the macros for the sensor module.

Table 5-56 List of Macros

File Name	Macro Name	Defined Value	Remarks
r_motor_sensor_api.h	SENSOR_MR_LOOP_POSITION	0	Makes calculations required for position control
	SENSOR_MR_LOOP_SPEED	1	Makes calculations required for speed control
r_motor_sensor_mr_digital.h	SENSOR_MR_CNTAVG	4	Number of pulse intervals required to average the speed
	SENSOR_MR_NUMB_OF_TIME	2	Maximum number of digital output pulse signal interrupts that can be processed per control interval
	SENSOR_MR_HS_CHANGE_RPM	(SENSOR_MR_NUMB_OF_TIME * 60) / (MOTOR_COMMON_CTRL_PERIOD * MOTOR_COMMON_SENSOR_MR_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_MR_HS_CHANGE_RAD	(SENSOR_MR_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_MR_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_MR_HS_CHANGE_MARGIN_RAD	SENSOR_MR_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_MR_PS_CALC_DIGITAL_INTERRUPT	0	Position and speed detection by using digital output pulse signal interrupts
	SENSOR_MR_PS_CALC_CTRL_INTERRUPT	1	Position and speed detection by using control interval interrupts
	SENSOR_MR_ZEROSPEED_PW	20000000	Pulse counter value to be judged as zero speed at low speed
	SENSOR_MR_ANGLE_DIFF	(MTR_TWOPI / (float)MOTOR_COMMON_SENSOR_MAG_CPR)	Angle of one pulse
	SENSOR_MR_CALC_AVG	(SENSOR_MR_CNTAVG * SENSOR_CFG_DIGITAL_TIMER_FREQ * 1000000.0f)	Constant used for recovering angle information from pulse

File Name	Macro Name	Defined Value	Remarks
r_motor_sensor_mr_digital.h	SENSOR_MR_HS_SPEED_LPF_PARAM	0.1f	LPF constant for speed calculation
	SENSOR_MR_PS_CALC_CHANGE_CNT	8	Number of times judgment is made at switchover between low speed and high speed
r_motor_sensor_mr_spi.h	SENSOR_MR_CMD_READ_REG	0x02	Reading of the magnet sensor register values
	SENSOR_MR_CMD_WRITE_REG	0x03	Writing of the magnet sensor register values
	SENSOR_MR_CMD_STRM_ANGL	0x05	Angle acquisition setting of the magnet sensor (streaming)
	SENSOR_MR_DT_DUMMY	0xFF	Dummy data for reading data by streaming
	SENSOR_MR_DT_DUMMY32	0xFFFFFFFF	Dummy data for reading data by streaming
	SENSOR_MR_SPI_ANGLE_DATA_MAX	65535.0f	Maximum angle value for the SPI
	SENSOR_MR_SPI_SCALING	(MTR_TWOPI / SENSOR_MR_SPI_ANGLE_DATA_MAX)	Scaling of the angle acquired for the SPI
	SENSOR_MR_SPI_RXBUFS	30	Buffer size for DTC
r_motor_sensor_mr.h	SENSOR_MR_ANGLE_ADJ_TIME	512	Adjustment value for the rotor pull-in time at startup by forced excitation (position/speed interrupt interval × adjustment value = pull-in time)
	SENSOR_MR_ANG_LIMIT_INACTIVE	0	Angle limit disabled
	SENSOR_MR_ANG_LIMIT_ACTIVE	1	Angle limit enabled
	SENSOR_MR_CARIB_SPEED_RPM	(6.0f)	Speed correction value [rpm]
	SENSOR_MR_CARIB_SPEED	((SENSOR_MR_CARIB_SPEED_RPM * MTR_TWOPI) / 60.0f * SENSOR_MR_POLE_PAIRS_INV * MOTOR_CFG_POLE_PAIRS)	Speed correction value [rad/s]
	SENSOR_MR_SPEED_DETECTION_CNT	1	Number of times that speed detection is skipped
	SENSOR_MR_ADC_GET_TIMING_CNT	(10.0f)	Number of times that ADC data acquisition is skipped

File Name	Macro Name	Defined Value	Remarks
r_motor_sensor_mr.h	SENSOR_MR_ADC_GET_RESO	((SENSOR_MR_CARIB_SPEED_RPM * 6.0f) * (1 / ((MOTOR_MCU_CFG_CARRIER_FREQ * 1000) / (MOTOR_MCU_CFG_INTR_DECIMATION + 1)) * SENSOR_MR_ADC_GET_TIMING_CNT))	ADC acquisition resolution
	SENSOR_MR_ADC_GET_CNT	(360.0f / SENSOR_MR_ADC_GET_RESO)	Number of times to acquire ADC
	SENSOR_MR_CALIB_CNT	(1 / (SENSOR_MR_CARIB_SPEED_RPM / 60) * ((MOTOR_MCU_CFG_CARRIER_FREQ * 1000) / (MOTOR_MCU_CFG_INTR_DECIMATION + 1)) / SENSOR_MR_ADC_GET_TIMING_CNT)	Number of times to perform calibration
	SENSOR_MR_ERROR_LIMIT	100	Error limit value
	SENSOR_MR_CALIB_ID	5	State of the d-axis current: Calibration in progress
	SENSOR_MR_CALIB_SPEED	5	Speed state: Calibration in progress
	SENSOR_MR_CALIB_INIT_SEQ	0	Initialization sequence of calibration
	SENSOR_MR_DATA_GET_ERR_CORRECT_SEQ	1	Correction sequence
	SENSOR_MR_CALIB_RESULT_JUDGE_SEQ	2	Judgement sequence
	SENSOR_MR_GET_SIGNAL_1ST	0	First data acquisition
	SENSOR_MR_GET_SIGNAL_2ND	1	Second data acquisition
	SENSOR_MR_GAIN_CORRECTION	2	Gain correction
	SENSOR_MR_GET_PHASE_CALIB_DATA	3	Acquisition of phase correction data
	SENSOR_MR_PHASE_CORRECTION	4	Phase correction
	SENSOR_MR_CALIB_FINISH	5	Correction completed
	SENSOR_MR_CALIB_PREPARED	0	Preparing for correction
	SENSOR_MR_CALIB_RUNNING	1	Correction in progress

File Name	**Macro Name**	**Defined Value**	**Remarks**
r_motor_sensor_mr.h	SENSOR_MR_CALIB_COMP_FINISH	2	Correction successfully completed
	SENSOR_MR_CALIB_ERROR_FINISH	3	Correction abnormally terminated
	SENSOR_MR_CALIB_DEF_POSITON	4	Misalignment of the motor and sensor in the initial position
	SENSOR_MR_OPENLOOP_INACTIVE	0	Open-loop control stopped
	SENSOR_MR_OPENLOOP_ACTIVE	1	Open-loop control in operation
	SENSOR_MR_SPEED_LPF_PARAM	(0.07f)	LPF constant for speed calculation
	SENSOR_MR_ADC_WAIT_LIMIT	10000	Limit value for A/D conversion wait time
	SENSOR_MR_ADC_REF_V	(5.0f)	Reference voltage for A/D conversion
	SENSOR_MR_ADC_REF_V_HALF	(SENSOR_MR_ADC_REF_V / 2.0f)	Intermediate voltage for A/D conversion
	SENSOR_MR_ADC_SCALING	(SENSOR_MR_ADC_REF_V / MOTOR_MCU_CFG_AD12BIT_DATA)	Scaling value for A/D conversion
	SENSOR_MR_CORRECT_ANGLE	(MTR_TWOPI / 8.0f)	Correction angle
	SENSOR_MR_POLE_PAIRS	(1.0f)	Number of pole pairs of the detection magnet
	SENSOR_MR_POLE_PAIRS_INV	(1.0f / SENSOR_MR_POLE_PAIRS)	Inverse of the number of pole pairs of the detection magnet

5.12.8 Adjustment and Configuration of Parameters

The initial values of the 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 magnet sensor digital output signals is to count signal edges. However, if you use a sensor with a low resolution, low speed cannot be calculated accurately due to the large ratio of 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 digital output signals to calculate the position and speed.

However, if this method is used with a high rotation speed or high-resolution sensor, a large number of interrupts by using digital output signals 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 digital output signal interrupts and calculation based on current control interrupts at a certain speed. Figure 5-36 shows an overview of this switchover. As shown in this figure, the speed calculation method changes from calculation based on digital output signal interrupts to calculation based on current control interrupts when the speed becomes high.

On the other hand, when the analog or SPI output magnet sensor or inductive sensor is used, the speed is calculated from difference from the previous angle value and control interval information based on current control interrupts.

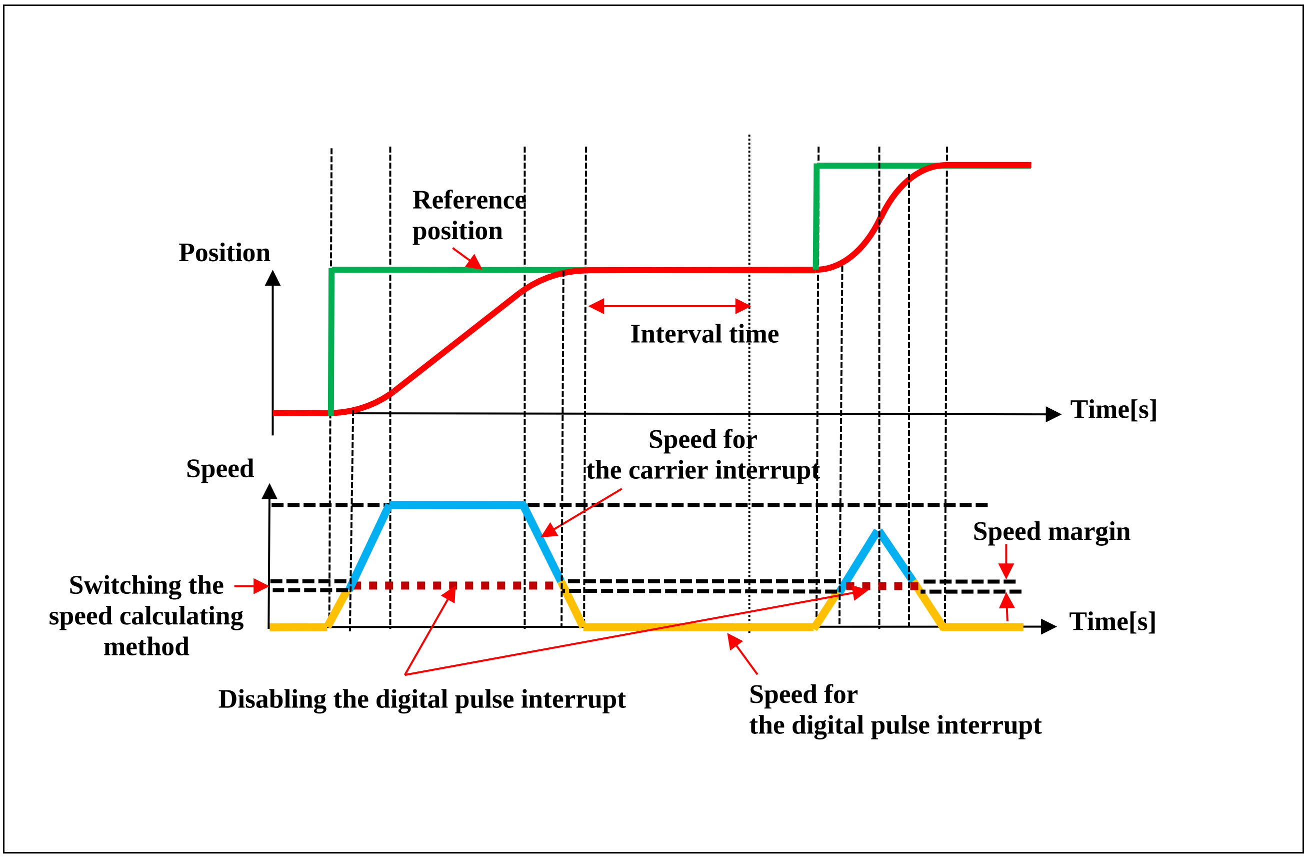
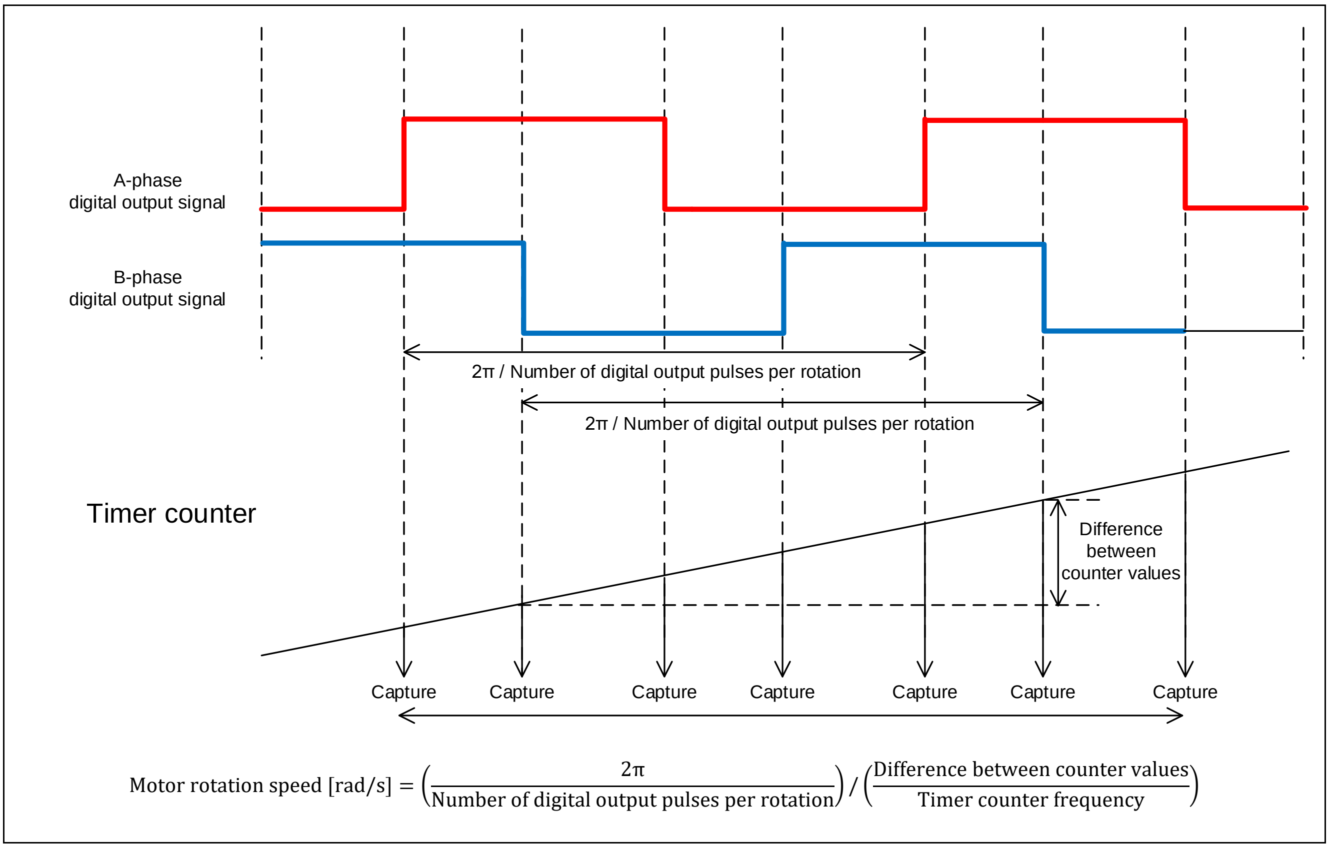


Figure 5-36 Switchover of the Method for Calculating the Position and Speed (Example)

5.12.10 Method for Calculating the Position and Speed by Using a Speed Sensor

(a) Speed calculation by using a digital output magnet sensor at low speed

Speed calculation at low speed is performed as shown in Figure 5-37

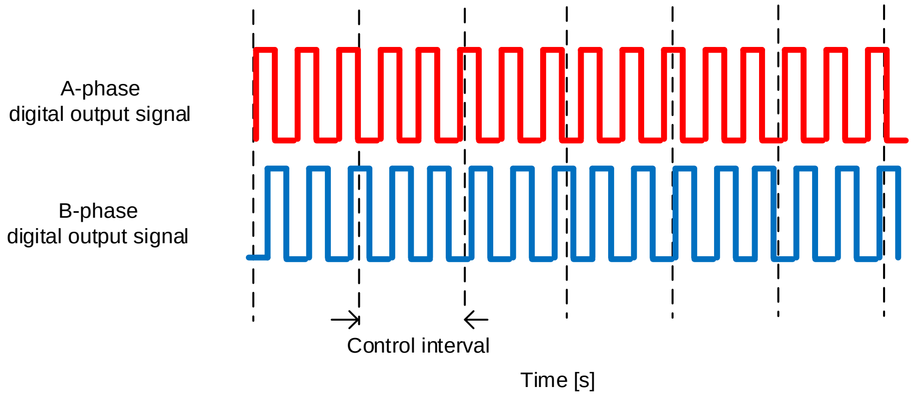


$$\text{Motor rotation speed [rad/s]} = \left(\frac{2\pi}{\text{Number of digital output pulses per rotation}}\right) / \left(\frac{\text{Difference between counter values}}{\text{Timer counter frequency}}\right)$$

Figure 5-37 Speed Calculation by Using the Digital Output Magnet Sensor at Low Speed

(b) Speed calculation by using a digital output magnet sensor at high speed

Speed calculation by using a digital output magnet sensor at high speed is performed as shown in Figure 5-38.



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

Figure 5-38 Speed Calculation by Using the Digital Output Magnet Sensor 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 program, and the Smart Configurator, which is required to access the microcontroller peripherals. Appropriately configuring the driver module allows you to use the differences in microcontroller function allocation and the specifications of the board to be used without modifying the motor module.

5.13.1 Functionality

Table 5-57 lists the functions of the driver module.

Table 5-57 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 an Smart Configurator function.
PWM duty setting	Sets the PWM duty value that is to be output to U-, V-, and W-phases via an Smart Configurator function.
PWM start/stop	Controls whether to start or stop PWM output via an Smart Configurator function.

5.13.2 Module Configuration Diagram

Figure 5-39 Driver Module Configuration Diagram shows the module configuration of the driver module.

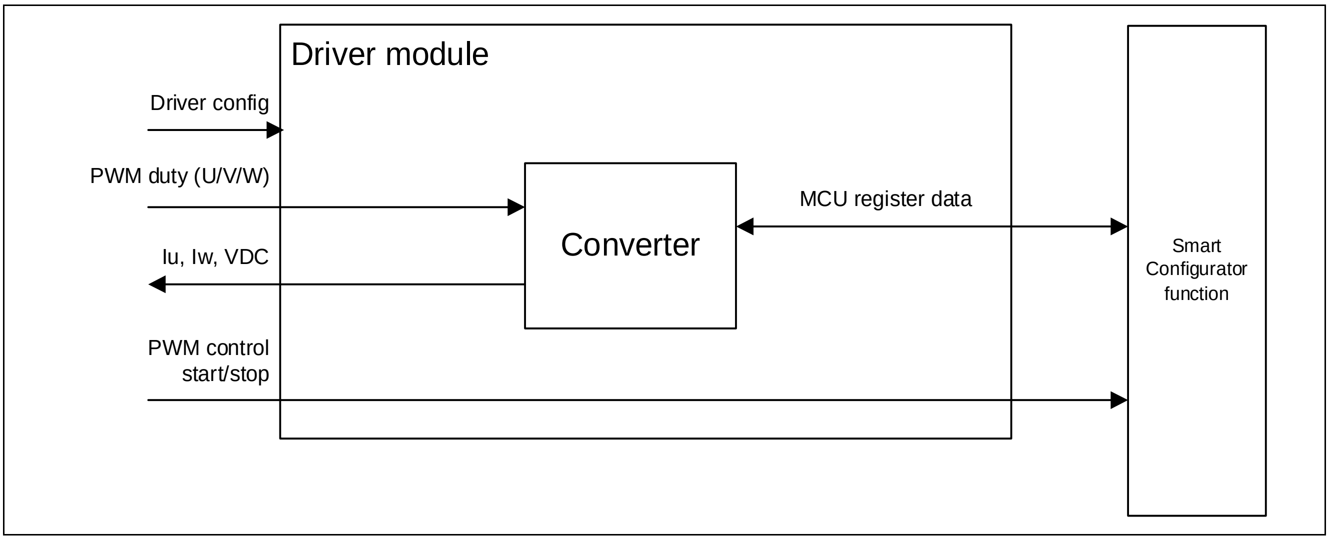


Figure 5-39 Driver Module Configuration Diagram

5.13.3 API

Table 5-58 lists and describes the API functions for the driver module.

Table 5-58 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 the 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 5-59 lists the configurations for the driver module. Set up the functions to be used and the necessary parameters. Table 5-60 shows the initial values for each microcontroller.

Table 5-59 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 5-60 List of Initial Values for Configurations

Macro Name	Set Value
DRIVER_CFG_FUNC_PWM_OUTPUT_START	R_Config_xxx_StartTimerCtrl (Smart Configurator function) *1 *2
DRIVER_CFG_FUNC_PWM_OUTPUT_STOP	R_Config_xxx_StopTimerCtrl (Smart Configurator function) *1 *2
DRIVER_CFG_FUNC_ADC_DATA_GET	R_Config_xxx_AdcGetConvVal (Smart Configurator function) *1 *2
DRIVER_CFG_FUNC_DUTY_SET	R_Config_xxx_UpdDuty (Smart Configurator function) *1 *2
INVERTER_CFG_ADC_REF_VOLTAGE	RX72M: 3.3f RX72T/RX66T/RX24T/RX24U/RX23T/RX13T: 5.0f
MOTOR_MCU_CFG_ADC_OFFSET	0x7FF

Notes: 1. For details about the functions shown in the "Set Value" column, see 5.14, Smart Configurator Settings.

2. When the Smart Configurator motor component is to be used, “xxx” is replaced with "MOTOR". When the motor component is not to be used, it is replaced with the module name used for the PWM.

5.13.5 Structure and Variable Information

Table 5-61 lists the structures for use with 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 5-61 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 number for one interval of the PWM counter (information for the duty setting)
	f4_pwm_dead_time_cnt	Count number 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 number for one interval of the PWM counter (information for the duty setting)
	f4_pwm_dead_time_cnt	Count number for the dead time (information for the duty setting)

5.13.6 Macro Definition

Table 5-62 lists the macros for the driver module.

Table 5-62 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 5-30.)	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_config_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 5-63 shows the settings.

Table 5-63 Example of Settings Specified in the Sample Program

Variable Name	Macro Name	File Name
*ADCDataGet	DRIVER_CFG_FUNC_ADC_DATA_GET	See Table 5-30.
*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, the 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 5-64 shows the clock settings.

Table 5-64 MCU Clock Settings

Clock	**Frequency**					
	RX13T	**RX23T**	**RX24T, RX24U**	**RX66T**	**RX72T**	**RX72M**
Main clock	8 MHz	10 MHz	10 MHz	8 MHz	10 MHz	24 MHz
System clock (ICLK)	32 MHz	40 MHz	80 MHz	160 MHz	200 MHz	240 MHz
Peripheral module clock (PCLKA)	32 MHz	40 MHz	40 MHz	80 MHz	100 MHz	120 MHz
Peripheral module clocks (PCLKB/PCLKC/PCLKD)	32 MHz/ - / 32 MHz	40 MHz/ - / 40 MHz	40 MHz/ 80 MHz/ 40 MHz	40 MHz/ 160 MHz/ 40 MHz	50 MHz/ 200 MHz/ 50 MHz	60 MHz/ 60 MHz/ 60 MHz
External bus clock (BCLK)	-	-	-	40 MHz	50 MHz	120 MHz
Flash memory interface clock (FCLK)	32 MHz	20 MHz	20 MHz	40 MHz	50 MHz	60 MHz
IWDTCLK	15 kHz			120 kHz		

5.14.2 Component Settings

Table 5-65 lists the components used and the functions allocated to the components.

Table 5-65 Smart Configurator Components and Their Functions

Function	Component					
	RX13T	RX23T	RX24T, RX24U	RX66T	RX72T	RX72M
3-phase PWM output, A/D conversion of current detection	Config_MOTOR					Config_GPT0/1/2 Config_S12AD0 Config_ELC
A/D conversion processing (inverter bus voltage detection, command voltage detection for the board UI)	Config_S12AD0		Config_S12AD2		Config_S12AD0	Config_S12AD1
A/D conversion processing for analog output	Not supported		Config_S12AD0	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_MTU0			Config_GPT3		
Independent watchdog timer	Config_IWDT					
Phase counting for digital output	Config_MTU0					
Phase counting for digital output	Config_MTU1					
SPI communications for SPI output	Not supported	Config_RSPI0				Config_RSPI2
DTC setting for SPI output	Not supported	Config_DTC				
Overcurrent detection	Config_POE					Config_POEG

5.14.3 Interrupts

Table 5-66 shows the information about the interrupts used for the MCUs that use the motor component. Table 5-67 shows the information about the interrupts used for the MCU that does not support the motor component (the RX72M).

Table 5-66 List of Interrupts (with the Motor Component)

Component	Interrupt Function	Description
Config_MOTOR	r_Config_MOTOR_ad_interrupt	A/D conversion end interrupt Interrupt level: 12 Multiple interrupt: Enabled
Config_S12AD0/1/2	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	Interrupt for digital output Interrupt level: 13 Multiple interrupt: Disabled
Config_MTU1	None	None
Config_RSPI0	r_Config_RSPI0_transmit_interrupt r_Config_RSPI0_receive_interrupt r_Config_RSPI0_error_interrupt r_Config_RSPI0_idle_interrupt	Interrupt for SPI communications Interrupt level: 15 Multiple interrupt: Disabled
Config_DTC	None	None
Config_POE	r_Config_POE_oei1_interrupt	Hardware overcurrent interrupt Interrupt level: 15 Multiple interrupt: Disabled

Table 5-67 List of Interrupts (without the Motor Component)

Component	Interrupt Function	Description
Config_ELC	None	None
Config_S12AD0	r_Config_S12AD0_interrupt	A/D conversion end interrupt Interrupt level: 10 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_GPT1	None	None
Config_GPT2	None	None
Config_GPT3	None	None
Config_IWDT	None	None
Config_MTU0	r_Config_MTU0_tgib0_interrupt	Interrupt for digital output Interrupt level: 13 Multiple interrupt: Disabled
Config_MTU1	None	None
Config_RSPI2	r_Config_RSPI2_transmit_interrupt r_Config_RSPI2_receive_interrupt r_Config_RSPI2_error_interrupt r_Config_RSPI2_idle_interrupt	Interrupt for SPI communications Interrupt level: 15 Multiple interrupt: Disabled
Config_DTC	None	None
Config_POEG	r_Config_POEG_poeggci_interrupt	Hardware overcurrent interrupt Interrupt level: 15 Multiple interrupt: Disabled

5.14.4 Details of User Codes

Table 5-68 lists the functions that are created in the user code area.

Table 5-68 List of Functions in the User Area

Component	Function	Description
Config_PORT	R_Config_PORT_GetSW1	Acquires the state of SW1.
	R_Config_PORT_GetSW2	Acquires the state of SW2.
	R_Config_PORT_Led1_on	Turns on LED1.
	R_Config_PORT_Led2_on	Turns on LED2.
	R_Config_PORT_Led3_on	Turns on LED3.
	R_Config_PORT_Led1_off	Turns off LED1.
	R_Config_PORT_Led2_off	Turns off LED2.
	R_Config_PORT_Led3_off	Turns off LED3.
	R_Config_PORT_HallSignalGet	Detects the Hall signal.
Config_GPT3 Config_MTU0	R_Config_xxxx_TcntGet*[1]	Acquires the timer counter value.
	R_Config_xxxx_SpeedCalcTimerStart*[1]	Starts the free-run timer for speed measurement.
Config_MTU0	R_Config_MTU0_InterruptEnable	Enables an interrupt for digital output.
	R_Config_MTU0_InterruptDisable	Disables an interrupt for digital output.
Config_MTU1	R_Config_MTU1_TcntSet	Sets the MTU1 timer counter.
	R_Config_MTU1_TcntGet	Reads the value from the MTU1 timer counter.
Config_RSPI0 Config_RSPI2	R_Config_xxxx_DtcBufferSet*[1]	Sets the DTC buffer
	R_Config_xxxx_AngleGet*[1]	Acquires an angle.
	R_Config_xxxx_TxBufferSet*[1]	Sets the 16-bit data buffer for transmission.
	R_Config_xxxx_TxBuffer2Set*[1]	Sets the 32-bit data buffer for transmission.
	R_Config_xxxx_RxBufferGet*[1]	Sets the 16-bit data buffer for reception.
	R_Config_xxxx_RxBuffer2Get*[1]	Sets the 32-bit data buffer for reception.
	R_Config_xxxx_StreamCommand*[1]	Sets the magnet sensor streaming mode.
Config_GPT0*[2]	R_Config_GPT0_StartTimerCount	Starts the counting of the PWM timer.
	R_Config_GPT0_StopTimerCount	Stops the counting of the PWM timer.
	R_Config_GPT0_StartTimerCtrl	Enables PWM output.
	R_Config_GPT0_StopTimerCtrl	Disables PWM output.
	R_Config_GPT0_UpdDuty	Writes data to the PWM duty setting register.
	R_Config_GPT0_StartAD	Enables the A/D conversion start and end interrupts.
	R_Config_GPT0_StopAD	Disables the A/D conversion stop and end interrupts.
	R_Config_GPT0_AdcGetConvVal	Acquires the A/D conversion value.

Notes: 1. The xxx portion is replaced with the name of the component to be used.
2. Set for only the products without the motor component.

5.14.5 Pin Settings

Table 5-69 shows the pin interface information.

Table 5-69 Pin Interfaces

Function	**RX13T**	**RX23T**	**RX24T, RX24U**	**RX66T**	**RX72T**	**RX72M**
Measurement of the inverter bus voltage	P46/AN006	P43/AN003	P64/AN204	P62/AN208	P63/AN209	P00/AN118
Pin for inputting the position/speed command value (analog value)	P47/AN007	P10/AN017	P53/AN209	P21/AN217	P43/AN003	P01/AN119
START/STOP toggle switch	PB5	P91	P80	P80	P35	PC5
ERROR RESET push-button switch	PB4	P92	P81	P81	PA0	PC3
LED1 control	PD6	P00	PA2	PE3	PC5	P80
LED2 control	PD4	P01	PA1	PB7	PC6	PK2
LED3 control	PB3	P31	PD7	PB3	P34	P76
Measurement of the U-phase current	P40/AN000	P40/AN000	P44/AN100	P40/AN000	P40/AN000	P40/AN000
Measurement of the W-phase current	P42/AN002	P42/AN002	P46/AN102	P42/AN002	P42/AN002	P42/AN002
PWM output (U_p)	P71/MTIOC3B	P71/MTIOC3B	P71/MTIOC3B	P71/MTIOC3B	P71/MTIOC3B	P23/GTIOC0A
PWM output (V_p)	P72/MTIOC4A	P72/MTIOC4A	P72/MTIOC4A	P72/MTIOC4A	P72/MTIOC4A	P22/GTIOC1A
PWM output (W_p)	P73/MTIOC4B	P73/MTIOC4B	P73/MTIOC4B	P73/MTIOC4B	P73/MTIOC4B	P21/GTIOC2A
PWM output (U_n)	P74/MTIOC3D	P74/MTIOC3D	P74/MTIOC3D	P74/MTIOC3D	P74/MTIOC3D	P17/GTIOC0B
PWM output (V_n)	P75/MTIOC4C	P75/MTIOC4C	P75/MTIOC4C	P75/MTIOC4C	P75/MTIOC4C	P87/GTIOC1B
PWM output (W_n)	P76/MTIOC4D	P76/MTIOC4D	P76/MTIOC4D	P76/MTIOC4D	P76/MTIOC4D	P86/GTIOC2B
SPI/communications (MOSI)	Not supported	P23/MOSIA	PD2/MOSIA	PD2/MOSIA	P21/MOSIA	PL1/MOSIC
SPI/communications (MISO)	Not supported	P22/MOSIA	PD1/MISOA	PD1/MISOA	P22/MISOA	PD2/MISOC
SPI/communications (CS port)	Not supported	PD6	PD6	PD6	P26	PJ2
SPI/communications (clock)	Not supported	P24/RSPCKA	PD0/RSPCKA	PD0/RSPCKA	P20/RSPCKA	PL0/RSPCKC
Sine input for analog output	Not supported	Not supported	P40/AN000	P44/AN100	PH5/AN104	PD4/AN112
Cosine input for analog output	Not supported	Not supported	P42/AN002	P46/AN102	PH6/AN105	PD5/AN113
A-phase input for digital output	PB1/MTCLKA	P33/MTCLKA	P33/MTCLKA	P33/MTCLKA	PA7/MTCLKA	P24/MTCLKA
B-phase input for digital output	PB0/MTCLKB	P32/MTCLKB	P32/MTCLKB	P32/MTCLKB	PA6/MTCLKB	P25/MTCLKB
PWM emergency stop input when an overcurrent is detected	PE2/POE10#	P70/POE0#	P70/POE0#	P70/POE0#	P70/POE0#	PC4/GTETRGC

6. Vector Control Algorithm

6.1 Analysis Model of a Permanent Magnet Synchronous Motor

The voltage equation of a permanent magnet synchronous motor that has sinusoidal magnetic flux distribution as shown in Figure 6-1 can be represented as below.

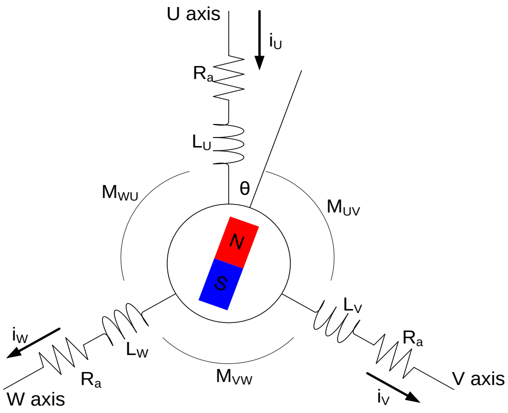


Figure 6-1 Conceptual Diagram for a 3-Phase Permanent Magnet Synchronous Motor

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

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

v_u, v_v, v_w: Armature voltages of each phase

i_u, i_v, i_w: Armature currents of each phase

ϕ_u, ϕ_v, ϕ_w: Armature interlinkage magnetic flux of each phase

R_a: Armature resistance of each phase

p: Differential operator

L_u, L_v, L_w: Self-inductances of each phase

M_{uv}, M_{vw}, M_{wu}: Mutual inductances between the phases

ψ: Maximum value of the armature interlinkage magnetic flux according to a permanent magnet

θ: Lead angle of a permanent magnet (rotor) from the U phase

6.2 d-q Axis Model of a Permanent Magnet Synchronous Motor

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

In the two-phase (d and q) coordinate system, the d axis is set in the direction of the magnetic flux (to the north pole) of the permanent magnet of the rotor, and the q axis is set at the lead angle (θ) of 90 degrees in the positive direction from the d axis. The following conversion matrix is used to obtain the voltage equation of the permanent magnet synchronous motor seen from the d-q coordinate system.

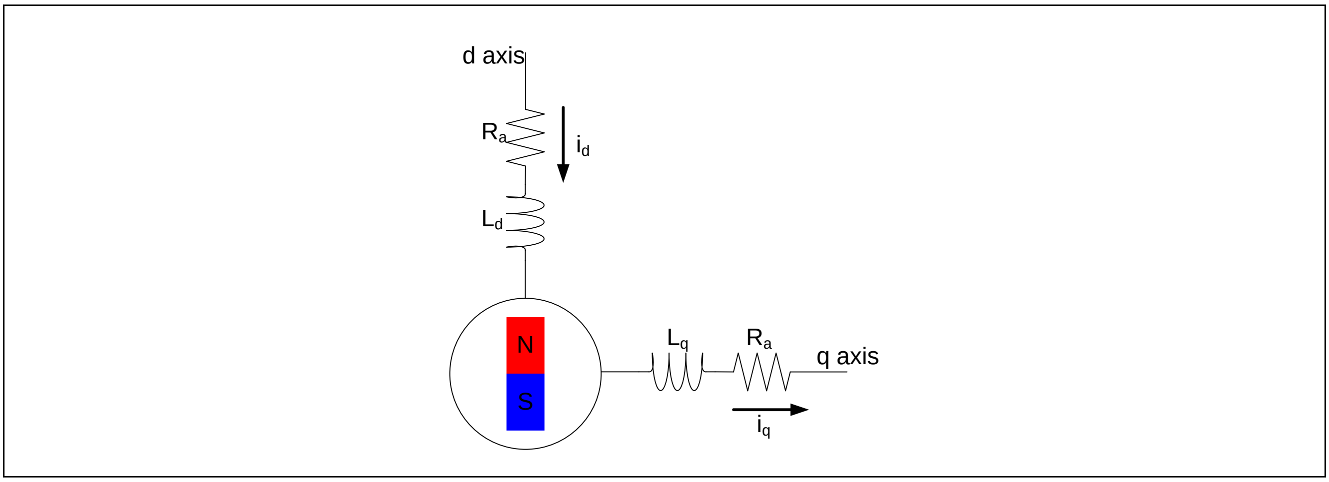


Figure 6-2 Conceptual Diagram for a Two-Phase DC Motor

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

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

From the coordinate transformation described on the previous page, the voltage equation in the d-q coordinate system can be represented as follows:

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

v_d, v_q: Armature voltages of the d and q axes

i_d, i_q: Armature currents of the d and q axes

R_a: Armature resistance of each phase

ω: Angular speed

L_d, L_q: Self-inductances of the d and q axes

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

ψ_a: Effective value of the armature interlinkage magnetic flux according to a permanent magnet

$$\psi_\mathrm{a} = \sqrt{\frac{3}{2}}\psi$$

As above, an alternating current that flows through a stationary three-phase stator can be assumed to be a direct current that flows through a two-phase stator that rotates in synchronization with the permanent magnet of the rotor.

The magnitude of torque generated in the motor is obtained from the cross product of the current vector and armature interlinkage magnetic flux as follows. In the following expression, the first term on the right side is referred to as the magnet torque and the second term on the right side is referred to as the reluctance torque.

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

T: Motor torque, P_n: Number of pole pairs

A motor having no difference in inductance between the d and q axes is called a "motor without saliency". In this case, the torque increases in proportion to the q-axis current because the reluctance torque is 0. Therefore, the q-axis current is sometimes referred to as the torque current. On the other hand, the d-axis current is sometimes referred to as the excitation current because it behaves as if the magnitude of the magnetic flux of the permanent magnet were changed according to d-axis current.

6.3 Vector Control System and Controller

The following shows the block diagram for the overall position control system.

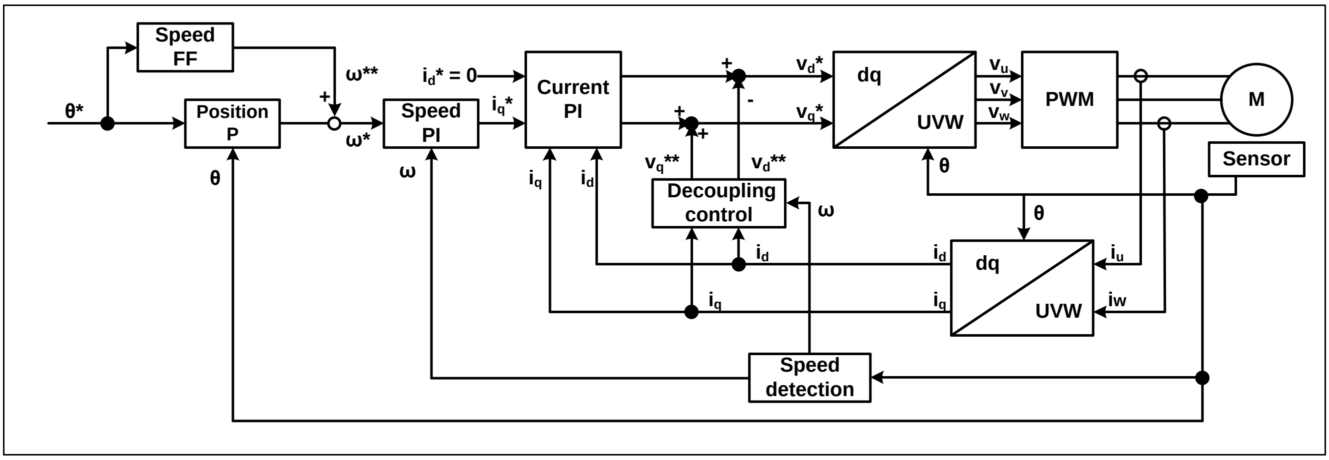


Figure 6-3 System Block Diagram of Vector Control (Position Control)

As shown in Figure 6-3, the position control system consists of the position control system, speed control system, and current control system. The speed and current control systems are implemented by using a general PI controller. The position control system is implemented by using the P controller and the feed-forward controller for the speed. Therefore, the gain of each controller must be designed appropriately in order to achieve the desired control characteristics.

Also, the decoupling control block in the system block diagram feeds the inductive voltages (v_d^{**} and v_q^{**}), which are generated when the motor rotates, forward to the command voltage of each phase. For v_d^{**} and v_q^{**}, see the expressions below. This achieves a highly responsive speed control system and enables independent control of the d and q axes.

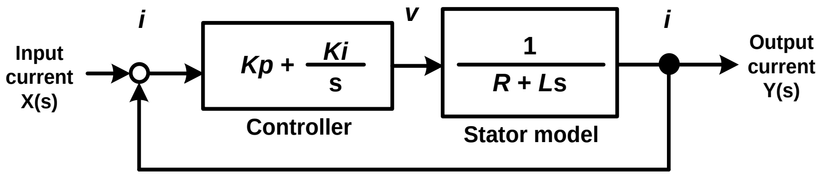
$$v_d^{**} = -\omega L_d i_q$$

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

6.3.1 Design of the Current Control System

This section describes how to model the current control system from a viewpoint of the electrical characteristics of the motor. A stator coil can be represented with the resistance R and inductance L. Therefore, the stator model of the motor can be represented as $\frac{1}{R+Ls}$, the transfer function of a general RL series circuit.

The controller can be represented by using PI control. The current control system can be represented as a feedback control system like the one shown in Figure 6-4.



R: Resistance of the stator coil $[\Omega]$
L: Inductance of the stator coil $[\mathrm{H}]$

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

Figure 6-4 Model of the Current Control System

First, design the PI control gains of the current control system on the assumption that the values of R and L for the motor stator are known.

The closed-loop transfer function of the current control lag system can be obtained as follows:

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

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

Also, the general expression of a second-order lag system having the zero point can be represented as follows:

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

Additionally, in coefficient comparison between the transfer function of the current control system and the general expression of the second-order lag system having the zero point, the relationships shown in the following expressions can be 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, damping ratio ζ, and zero-point frequency ω_z can be represented as follows:

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

From this, the current PI control gains $K_{p\_current}$ and $K_{i\_current}$ can be represented by the following expressions:

$$K_{p\_current}=2\zeta_{CG}\omega_{CG}L-R, \qquad K_{i\_current}=K_{p\_current}a=\omega_{CG}^2 L$$

ω_{CG}: Natural frequency for the current control system

ζ_{CG}: Damping ratio for the current control system

Therefore, it is found that the PI control gains of the current control system can be designed with ω_{CG} and ζ_{CG}.

6.3.2 Design of the Speed Control System

This section describes how to model the speed control system from a viewpoint of the mechanical characteristics of the motor. From the motion equation of the rotation system, the torque expression of the mechanical system can be represented by the following expression:

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

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

On the other hand, the torque expression of the electrical system can be represented by the following expression if only the magnet torque is considered:

$$T = P_n \psi_a i_q$$

By using the above two torque expressions of the mechanical and electrical systems, the mechanical angular speed can be represented by the following expression:

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

ω_{mech}: Mechanical angular speed

Therefore, this is the motor model in the speed control system. Also, the controller can be represented by using PI control. The speed control system can be represented as a feedback control system like the one shown in Figure 6-5.

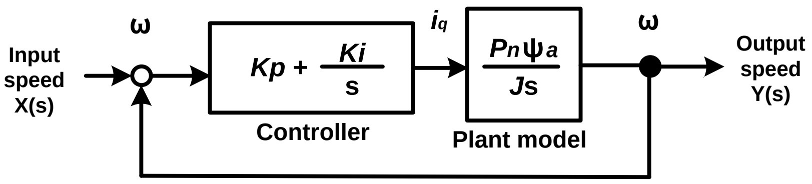


Figure 6-5 Model of the Speed Control System

Here, set the PI control gains of the speed control system on the assumption that the motor parameters P_n, ψ, J are known. First, obtain the transfer function of the system.

The closed-loop transfer function of the speed control system can be obtained as follows:

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

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

Also, the general expression of a second-order lag system having the zero point can be represented as follows:

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

In the same way as for the current control system, in coefficient comparison between the transfer function of the speed control system and the general expression of the second-order lag system having the zero point, the relationships shown in the following expressions 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, damping ratio ζ, and zero-point frequency ω_z can be represented 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} can be represented by the following expressions:

$$K_{p\_speed} = \frac{2\zeta_{SG}\omega_{SG} J}{P_n \psi_a}, \qquad K_{i\_speed} = K_{p\_speed} * a = \frac{\omega_{SG}^2 J}{P_n \psi_a}$$

ω_{SG}: Natural frequency for the speed control system

ζ_{SG}: Damping ratio for the speed control system

Therefore, it is found that the PI control gains of the speed control system can be designed with ω_{SG} and ζ_{SG}.

6.3.3 Design of the Position Control System

The controller of the position control system uses only proportional terms. To achieve faster response to input that is much larger than the speed command value, the responsiveness is improved in combination with feed-forward control for the speed. The following shows the block diagram of the position control system.

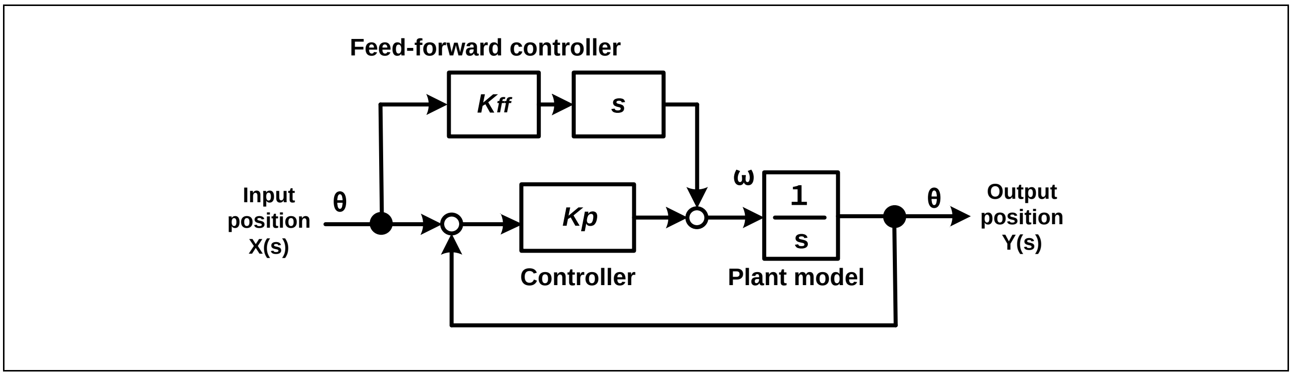


Figure 6-6 Model of the Position Control System

The position control system uses only P control. Therefore, to design the gain $K_{p\_position}$, use only the natural frequency for the position control system (ω_{PG}).

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

$$K_{p\_position} = \omega_{PG}$$

Also, to improve the speed responsiveness, feed-forward control for the speed is implemented.

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

Therefore, you can design the speed feed-forward control by using a fixed value and can design the P gain of the position by using the natural frequency ω_n.

7. Test Results

The test results provided in this chapter are for reference only and were obtained in the environment as described in 2.1, Test Environments.

7.1 Program Size

The following tables show the size of the sample program. In the optimization settings of the compiler, the optimization level is set to 2 (-optimize = 2) and the optimization method is set to the one that is code-size oriented (-size).

Table 7-1 Program Size (Digital output)

Memory	RX13T	RX23T	RX24T	RX24U	RX66T	RX72T	RX72M
ROM	24.5 KB	24.0 KB	24.8 KB	25.0 KB	29.1 KB	29.1 KB	30.8 KB
RAM	5.0 KB	5.1 KB	5.2 KB	5.2 KB	9.2 KB	9.1 KB	9.4 KB
Maximum value of the stack analysis result	356 B	356 B	352 B	352 B	356 B	352 B	352 B
Stack size setting value for IDE environment	1536 B	1536 B	1536 B	1536 B	5120 B	5120 B	5120 B

Table 7-2 Program Size (Analog output)

Memory	RX24T	RX24U	RX66T	RX72T	RX72M
ROM	27.4 KB	27.4 KB	31.8 KB	31.9 KB	33.8 KB
RAM	5.3 KB	5.3 KB	9.3 KB	9.2 KB	9.5 KB
Maximum value of the stack analysis result	360 B	360 B	360 B	360 B	360 B
Stack size setting value for IDE environment	1536 B	1536 B	5120 B	5120 B	5120 B

Table 7-3 Program Size (SPI output)

Memory	RX23T	RX24T	RX24U	RX66T	RX72T	RX72M
ROM	23.9 KB	24.7 KB	24.8 KB	29.0 KB	29.0 KB	30.7 KB
RAM	5.2 KB	5.3 KB	5.3 KB	9.3 KB	9.2 KB	9.5 KB
Maximum value of the stack analysis result	352 B	348 B	348 B	352 B	348 B	348 B
Stack size setting value for IDE environment	1536 B	1536 B	1536 B	5120 B	5120 B	5120 B

7.2 CPU Loading Rate

The following table shows the CPU processing time and loading rate for each control interval.

Table 7-4 Control Loop and CPU Loading Rate (Digital output)

MCU	Control Loop Type	Control Interval	Processing Time	CPU Loading Rate
RX72M	Current control loop	50 µs (no skip)	9.42 µs	18.8 %
	Speed/position control loop	500 µs	2.44 µs	0.5 %
RX72T	Current control loop	50 µs (no skip)	9.88 µs	19.8 %
	Speed/position control loop	500 µs	2.935 µs	0.6 %
RX66T	Current control loop	50 µs (no skip)	12.8 µs	25.6 %
	Speed/position control loop	500 µs	3.62 µs	0.7 %
RX24T	Current control loop	100 µs (1 skip)	35.65 µs	35.7 %
	Speed/position control loop	1000 µs	9.82 µs	1.0 %
RX24U	Current control loop	100 µs (1 skip)	35.4 µs	35.4 %
	Speed/position control loop	1000 µs	9.78 µs	1.0 %
RX23T	Current control loop	100 µs (1 skip)	69.0 µs	69.0 %
	Speed/position control loop	1000 µs	18.5 µs	1.9 %
RX13T	Current control loop	125 µs (1 skip)	101.4 µs	81.1 %
	Speed/position control loop	1000 µs	23.9 µs	2.4 %

Table 7-5 Control Loop and CPU Loading Rate (Analog output)

MCU	Control Loop Type	Control Interval	Processing Time	CPU Loading Rate
RX72M	Current control loop	50 µs (no skip)	10.6 µs	21.2 %
	Speed/position control loop	500 µs	2.54 µs	0.5 %
RX72T	Current control loop	50 µs (no skip)	13.64 µs	27.3 %
	Speed/position control loop	500 µs	3.01 µs	0.6 %
RX66T	Current control loop	50 µs (no skip)	18.5 µs	37.0 %
	Speed/position control loop	500 µs	3.77 µs	0.8 %
RX24T	Current control loop	100 µs (1 skip)	42.8 µs	42.8 %
	Speed/position control loop	1000 µs	9.6 µs	1.0 %
RX24U	Current control loop	100 µs (1 skip)	43.0 µs	43.0 %
	Speed/position control loop	1000 µs	9.74 µs	1.0 %

Table 7-6 Control Loop and CPU Loading Rate (SPI output)

MCU	Control Loop Type	Control Interval	Processing Time	CPU Loading Rate
RX72M	Current control loop	50 µs (no skip)	8.8 µs	17.6 %
	Speed/position control loop	500 µs	2.47 µs	0.5 %
RX72T	Current control loop	50 µs (no skip)	9.22 µs	18.4 %
	Speed/position control loop	500 µs	2.975 µs	0.6 %
RX66T	Current control loop	50 µs (no skip)	13.22 µs	26.4 %
	Speed/position control loop	500 µs	3.65 µs	0.7 %
RX24T	Current control loop	100 µs (1 skip)	33.9 µs	33.9 %
	Speed/position control loop	1000 µs	10.14 µs	1.0 %
RX24U	Current control loop	100 µs (1 skip)	33.3 µs	33.3 %
	Speed/position control loop	1000 µs	10.02 µs	1.0 %
RX23T	Current control loop	100 µs (1 skip)	64.1 µs	64.1 %
	Speed/position control loop	1000 µs	19.3 µs	1.9 %

7.3 Operation Waveforms

The waveforms during control by using the RX72T are given below and on the following page as the results of tests for reference.

Table 7-7 Measurement Conditions

Item	Value	Remarks
Frequency for the current control system	300 [Hz]	
Damping ratio for the current control system	1	
Frequency for the speed control system	12 [Hz]	
Damping ratio for the speed control system	1	
Frequency for the position control system	4 [Hz]	Valid only during position control.
Load	—	Conducted at no load.

Figure 7-1 shows the results of testing speed control.

Time/Div 100.00m Mode Auto Edge Rise Source EXT Level 0.00 Position 400.00m

Ready CPU : RX72T Serial : SCI6 PORT :COM4

Figure 7-1 Speed Control by Using the Magnet Sensor

Drive condition:

Rotation speed: Commanded speed 1000 [rpm]

Waveform information:

Yellow: Detected speed [rad/s] (200 rad/s/div.)
Orange: Commanded speed [rad/s] (200 rad/s/div.)
Red: q-axis current command value [A] (500 mA/div.)
Purple: q-axis current value [A] (500 mA/div.)
Pink: d-axis current command value [A] (1 A/div.)
White: d-axis current value [A] (1 A/div.)
Light green: U-phase voltage [V] (5 V/div.)
Horizontal axis: 100 ms/div.

Figure 7-2 shows the results of testing position control.

Figure 7-2 Position Control by Using the Magnet Sensor

Drive condition:

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

Waveform information:

Yellow: Detected speed [rad/s] (200 rad/s/div.)
Orange: Commanded speed [rad/s] (200 rad/s/div.)
Red: q-axis current command value [A] (500 mA/div.)
Purple: q-axis current value [A] (500 mA/div.)
Pink: d-axis current command value [A] (1 A/div.)
White: d-axis current value [A] (1 A/div.)
Turquoise: Angle information (mechanical angle) calculated from the sensor [rad] (20 rad/div.)
Blue: Position command value [rad] (20 rad/div.)
Horizontal axis: 100 ms/div.

8. Reference Materials

- Renesas Motor Workbench V.3.00 User's Manual (R21UZ0004)
- Evaluation System for BLDC Motor User's Manual (R12UZ0062)
- Smart Configurator User's Manual -- RX API Reference (R20UT4360)
- RX Smart Configurator User Guide -- CS+ (R20AN0470)
- RX Smart Configurator User Guide -- e^2 studio (R20AN0451)
- RX72M CPU Card with RDC-IC User’s Manual (R12UZ0098)
- RX72M Group User's Manual -- Hardware (R01UH0804)
- RX13T CPU Card User's Manual (R12UZ0051)
- RX13T Group User's Manual -- Hardware (R01UH0822)
- RX23T CPU Card User’s Manual (R20UT3698)
- RX23T Group User's Manual -- Hardware (R01UH0520)
- RX24T CPU Card User's Manual (R20UT3696)
- RX24T Group User's Manual -- Hardware (R01UH0576)
- RX24U CPU Card User's Manual (R12TU0018)
- RX24U Group User's Manual -- Hardware (R01UH0658)
- RX66T CPU Card User's Manual (R12UZ0028)
- RX66T Group User's Manual -- Hardware (R01UH0749)
- RX72T CPU Card User’s Manual (R12UZ0031)
- RX72T Group User's Manual -- Hardware (R01UH0803)

9. Revision History

Rev.	Date of Issue	Amendments	
		Page	**Point**
1.00	May 23.22	—	First edition issued

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/.