120-degree Conducting Control of a Permanent Magnet Synchronous Motor

for the Evaluation System for BLDC Motor

Introduction

This application note is intended to explain the software sample program that uses a microcontroller manufactured by Renesas to drive a permanent magnet synchronous motor with 120-degree conducting control. The target software for this application uses the Smart Configurator tool and the components required for motor control.

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

Target Device

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

- MCUs used:
- RX24U (R5F524UEADFB)
- RX24T (R5F524TAADFP)
- RX23T (R5F523T5ADFM)
- RX13T (R5F513T5ADFL)

Target Software

The following shows the target software for this application:

- RXxxx_ESB_SPM_120_CONDUCTION_CSP_V100 (IDE: CS+ edition)*[1]
- RXxxx_ESB_SPM_120_CONDUCTION_E2S_V100 (IDE: e^{2}studio edition)*[1]
  Evaluation System for BLDC Motor & Encoder vector control software for the RXxxx CPU card*[1]

Note: 1. The xxx portion is replaced by the name of the MCU to be used.

Contents

1. Overview

This application note is intended to explain the method of using the sample program that uses a microcontroller manufactured by Renesas to drive a permanent magnet synchronous motor with 120-degree conducting control. 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.

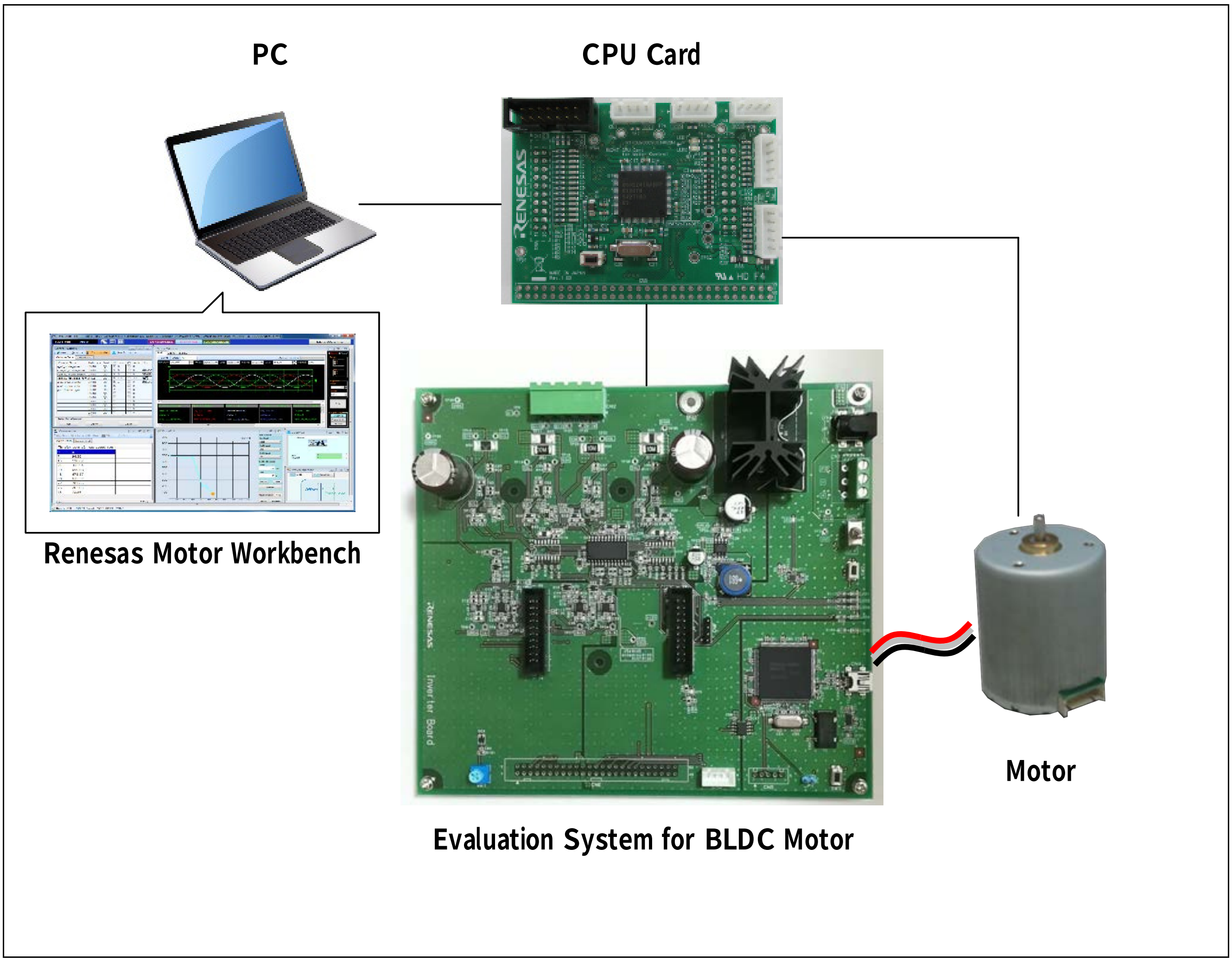


Figure 1-1 Operating environment of the sample program

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 is applicable to.

Table 2-1 Hardware development environment

Category	Product used
Microcontroller/CPU card product type	RX24U (R5F524UEADFB)/RTK0EMX590C02000BJ RX24T (R5F524TAADFP)/RTK0EM0009C03402BJ RX23T (R5F523T5ADFM)/RTK0EM0003C01202BJ RX13T (R5F513T5ADFL)/RTK0EMXA10C00000BJ
Inverter board	Evaluation System for BLDC Motor (RTK0EM0000B10020BJ) included Inverter board for 48 V 5 A BLDC
Motor	TG55L-KA (Manufactured by TSUKASA ELECTRIC CO., LTD.)
Sensor	Hall: TG55L-KA (Manufactured by TSUKASA ELECTRIC CO., LTD.) Sensorless: None

Table 2-2 Software development environment

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

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

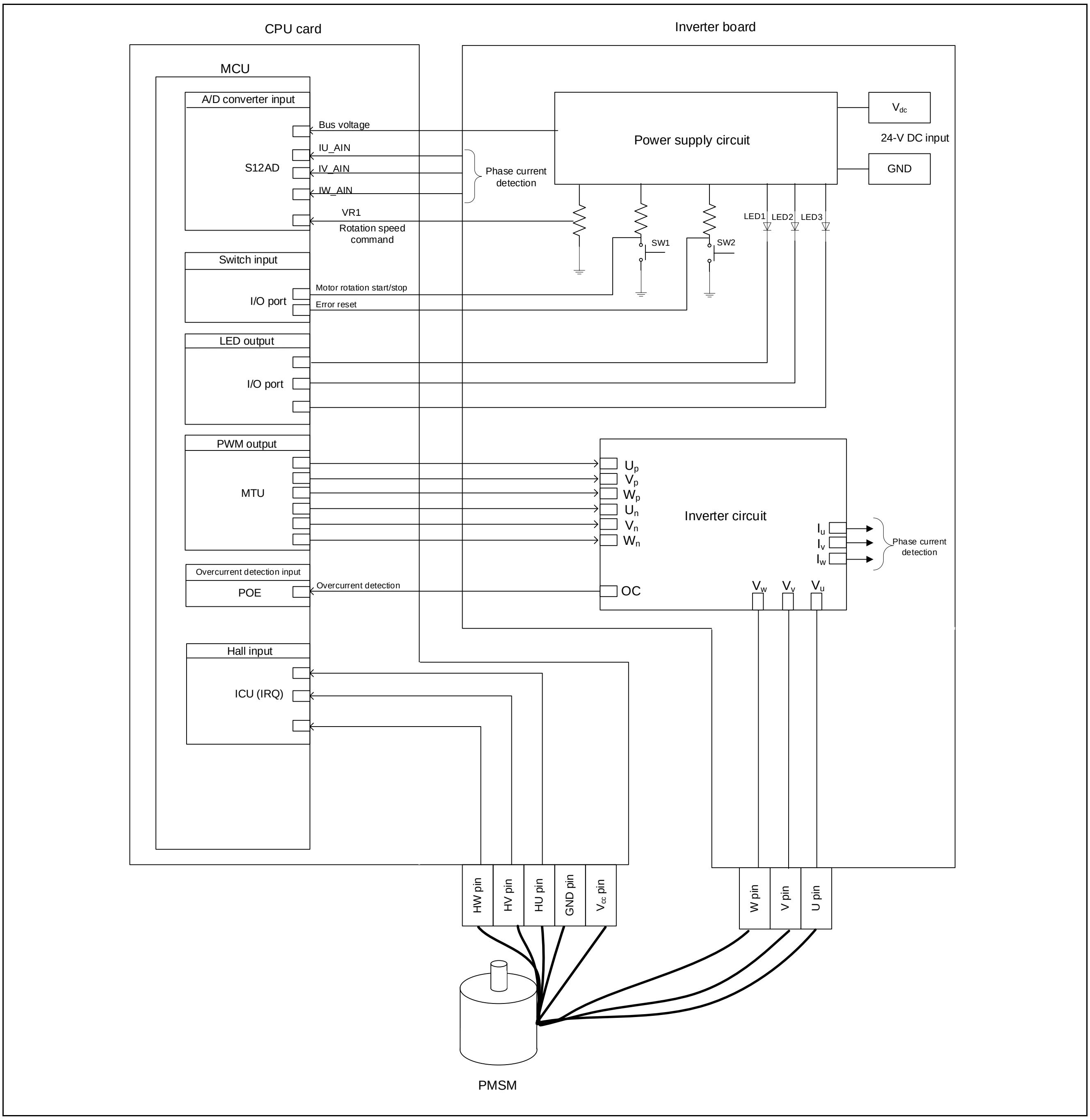


Figure 2-1 Hardware configuration diagram (with a hall sensor)

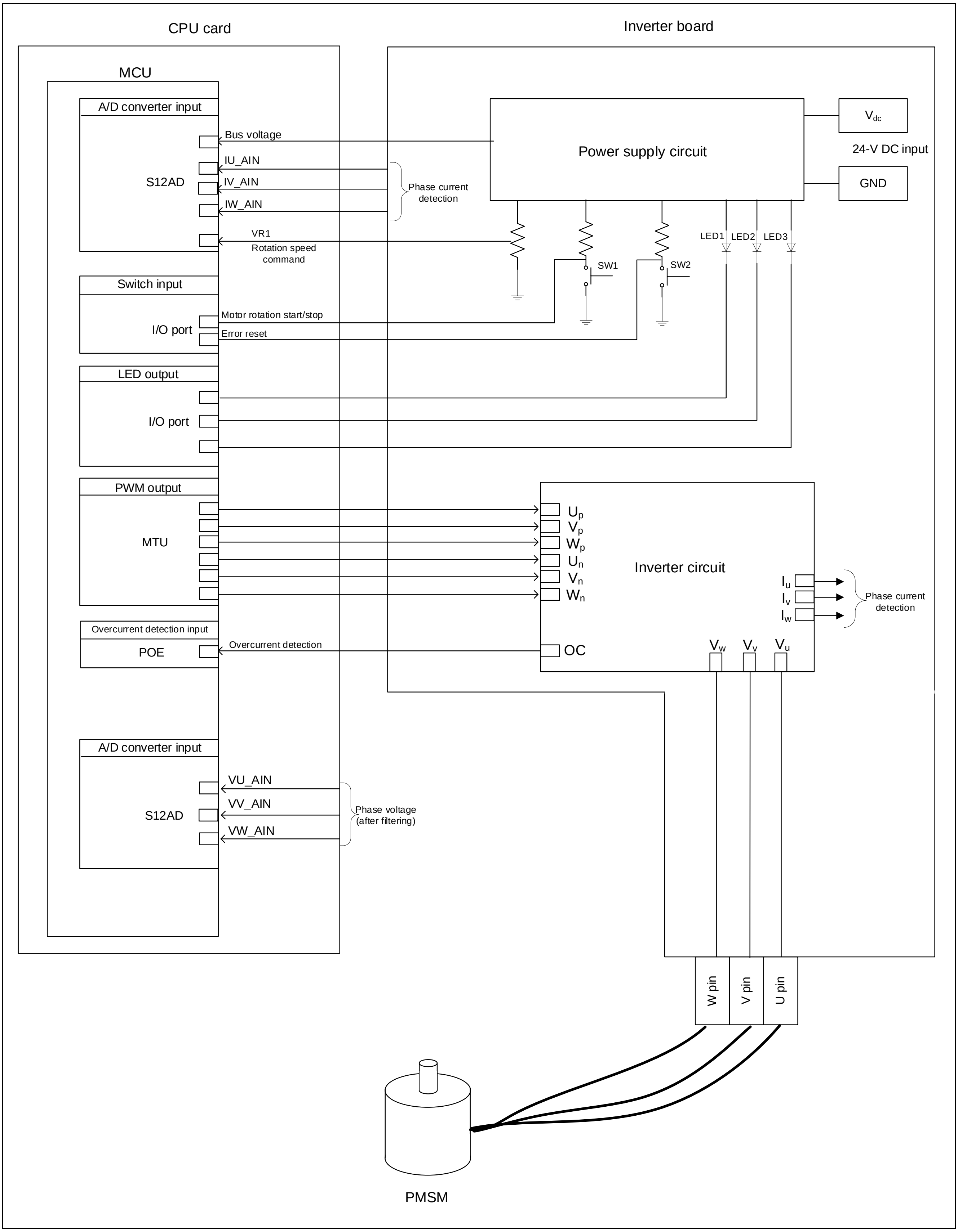


Figure 2-2 Hardware configuration diagram (sensorless)

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 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	Not used in this system
RESET	Push 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, Smart Configurator is used to configure the peripheral functions. For details, see 5.6.

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 V-phase current	S12AD
Measurement of the W-phase current	S12AD
Measurement of the U-phase voltage	S12AD
Measurement of the V-phase voltage	S12AD
Measurement of the W-phase voltage	S12AD
PWM output (U_p)/"Low" active	MTU
PWM output (V_p)/"Low" active	MTU
PWM output (W_p)/"Low" active	MTU
PWM output (U_n)/"High" active	MTU
PWM output (V_n)/"High" active	MTU
PWM output (W_n)/"High" active	MTU
Hall U-phase input	ICU (IRQ)
Hall V-phase input	ICU (IRQ)
Hall W-phase input	ICU (IRQ)
PWM emergency stop input when an overcurrent is detected	POE

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 Renesas Motor Workbench (RMW), see the "Renesas Motor Workbench User's Manual" (R21UZ0004).

3.1 Downloading and writing the sample program

After you have downloaded the sample program from our website, use an integrated development environment (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 special 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). Renesas Motor Workbench (RMW) can be downloaded from our website.

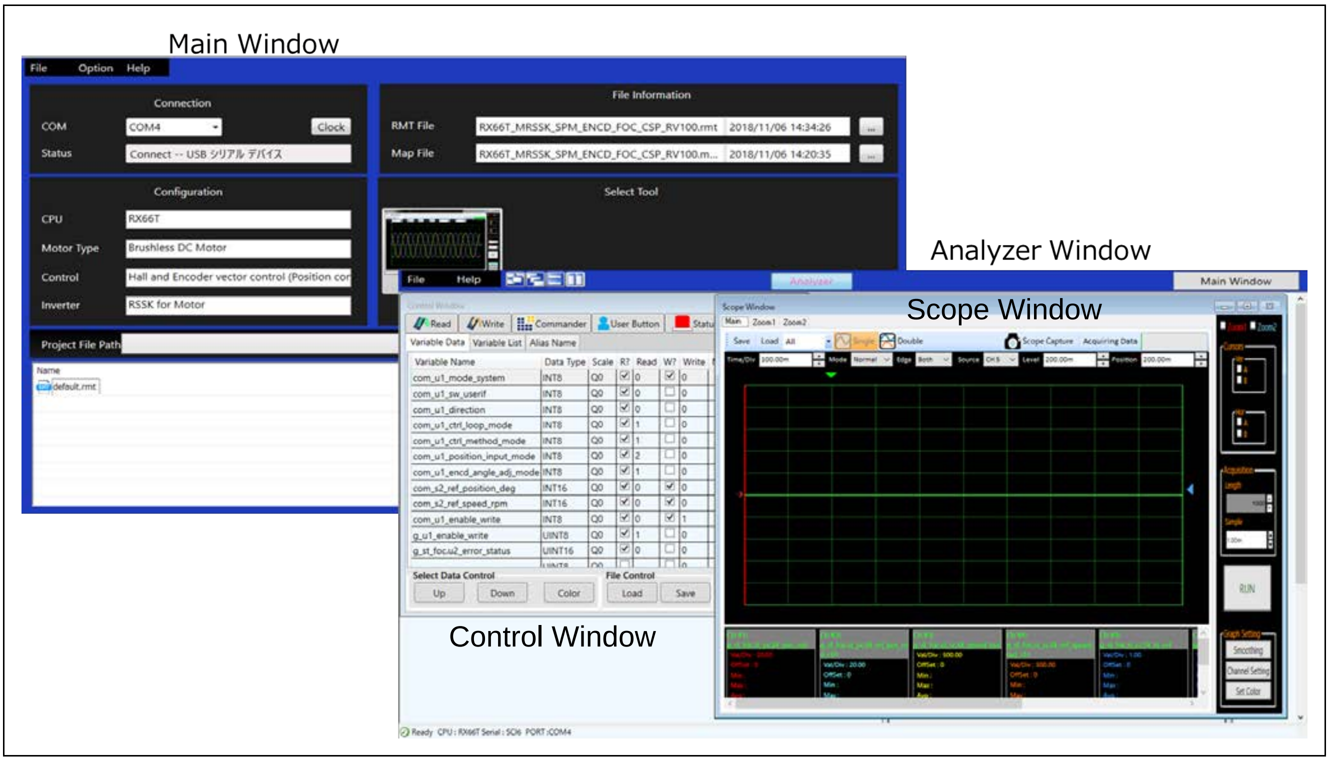


Figure 3-1 Windows of Renesas Motor Workbench

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

- Click the 

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 stores the environmental information that was manipulated or configured by using 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 and 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_s2_ref_speed_rpm (*)	int16_t	Speed command value (mechanical angle) [rpm]
com_u1_enable_write	uint8_t	Whether to enable rewrite 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 observed when the driving under 120-degree conducting control is evaluated. 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 5.1.4.

Table 3-2 List of main variables for speed control

Main encoder position/speed control variable name	Type	Description
g_st_120_conduction.u2_error_status	uint16_t	Error status
g_st_speed.f4_ref_speed_rad	float	Speed command value (mechanical angle) [rad/s]
g_st_speed.f4_speed_rad	float	Speed detection value (mechanical angle) [rad/s]
g_st_speed.f4_v_ref	float	Command voltage [V]

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) Start 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.

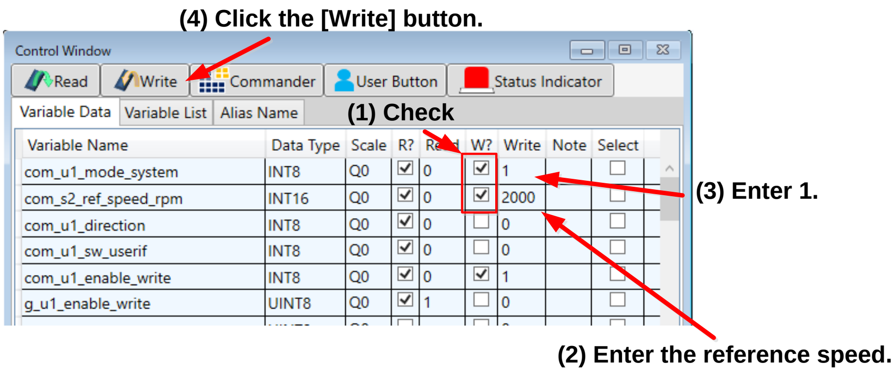


Figure 3-2 Procedure for starting rotation of the motor

(b) Stop the motor

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

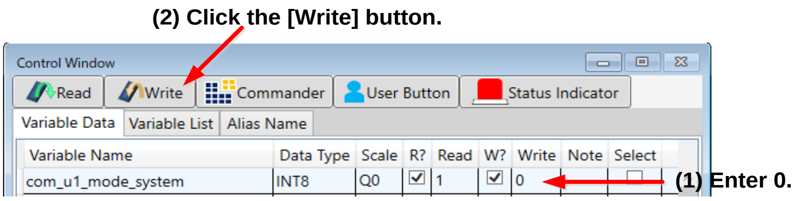


Figure 3-3 Procedure for stopping the motor

(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.

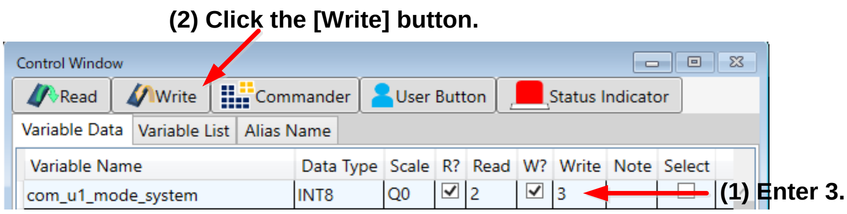


Figure 3-4 Procedure for handling an error

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.

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

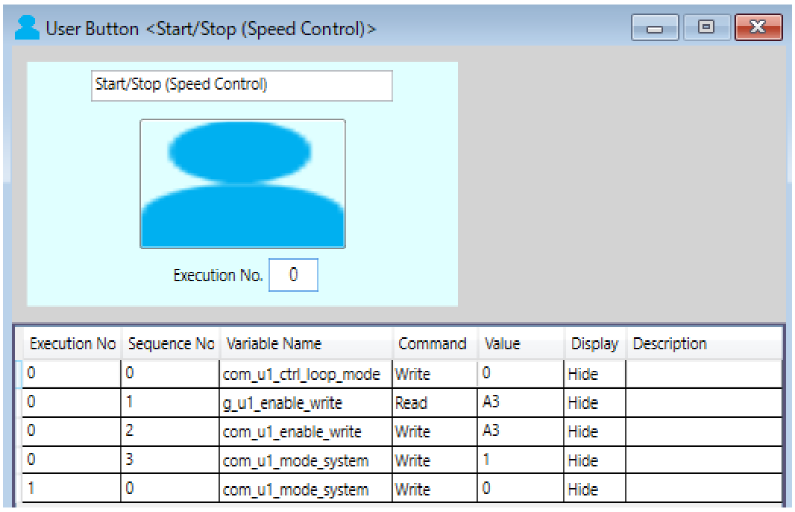


Figure 3-5 Starting or stopping motor rotation

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

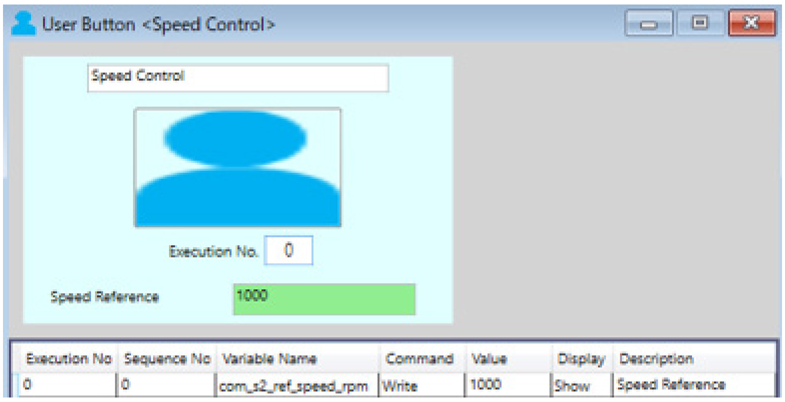


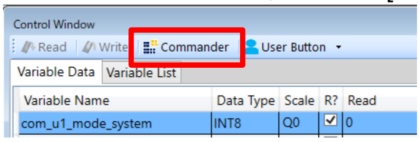
Figure 3-6 Changing the speed command

3.4.3 Operation example of the Commander function

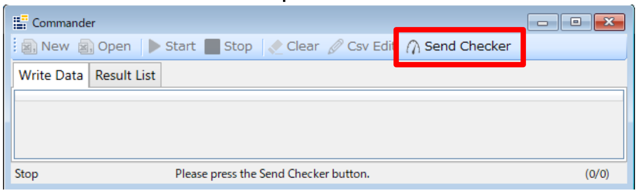
Using the Commander function to perform position control

Starting Commander:

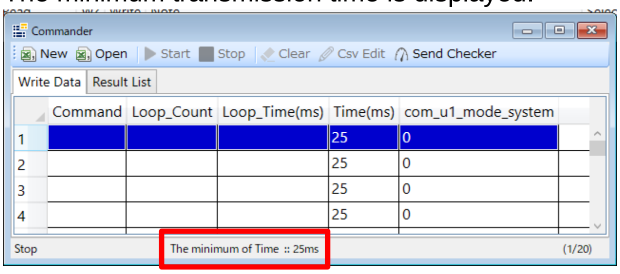
(1) In the Control Window, click the [Commander] button.



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

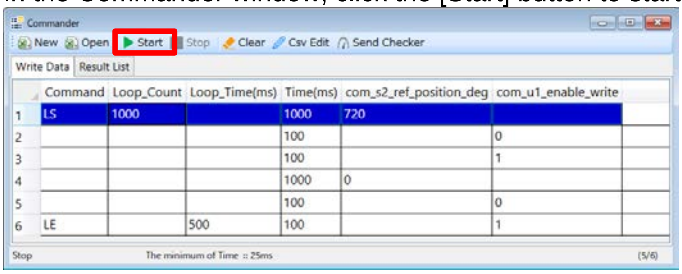


The minimum transmission time is displayed.



(3) Click the [Open] button to load "Position_test.csv". Place the system in the speed 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 speed control.

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



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 mark in the [W?] column is selected, and then enter "1" in the [Write] column. Click the [Write] button.

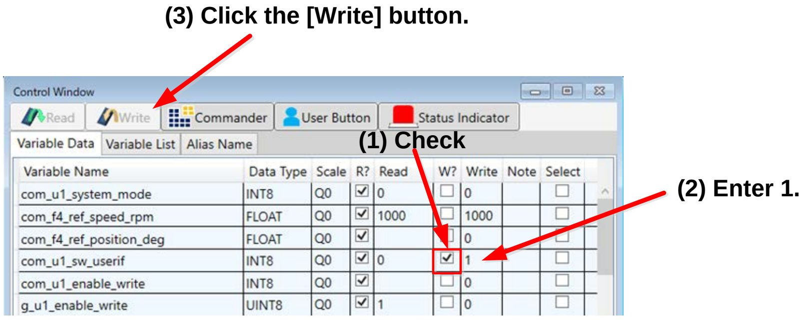


Figure 3-7 Procedure for switching the UI

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 the SW1 on the inverter board (via the Board UI). A general-purpose port is assigned to the 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 speed command value is determined by performing A/D conversion for the output value (analog value) of the VR1 on the inverter board. The VR1 value after A/D conversion is used as the 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 speed command value	CW	0 to 2650 [rpm]: 07FFH to 0000H
	CCW	0 to -2650 [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 120-degree conducting control software

Item	Description	
Control method	120-degree conducting control (first 60-degrees chopping)	
Starting/stopping motor control	Determined by the level of SW1 (“Low”: start control; “High”: stop) or input from RMW	
Rotor magnetic pole position detection	Hall sensor: Position detection by using a Hall sensor (every 60 degrees) Sensorless: Position detection using the induced voltage (every 60 degrees)	
Input voltage	24 V DC	
Carrier frequency (PWM)	20 [kHz], Carrier cycle: 50 [μs]	
Dead time	2 [μs]	
Control cycle (speed)	Hall sensor: • Every interrupt signal on both edges of a Hall sensor input signal Sensorless: • Zero-crossing is detected from the induced voltage for every carrier cycle. • When the pattern is switched, the PWM duty is set and conduction pattern is determined. Speed PI control is performed every 2 [ms].	
Speed command value management	Hall sensor: 550 to 2650 [rpm] for both CW and CCW Sensor less: 1000 to 2650 [rpm] for both CW and CCW	
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 be deactivated when any of the following conditions are met: 1. The currents of all phases exceed 0.89 [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 3000 [rpm] (checked at intervals of 50 [μs]). 5. In a hall sensor drive configuration, hall interrupt detection does not occur for 200 [ms]. 6. In a sensorless drive configuration, zero-crossing detection does not occur for 100 [ms]. 7. A Hall sensor pattern error has occurred. 8. An abnormal pseudo Hall sensor pattern (position information) is detected. When the overcurrent detection signal (POE) from an external circuit or an output short-circuit is detected, the PWM output pin is driven to high impedance.	

4.2 Software configuration

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

4.2.1 Overall configuration

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

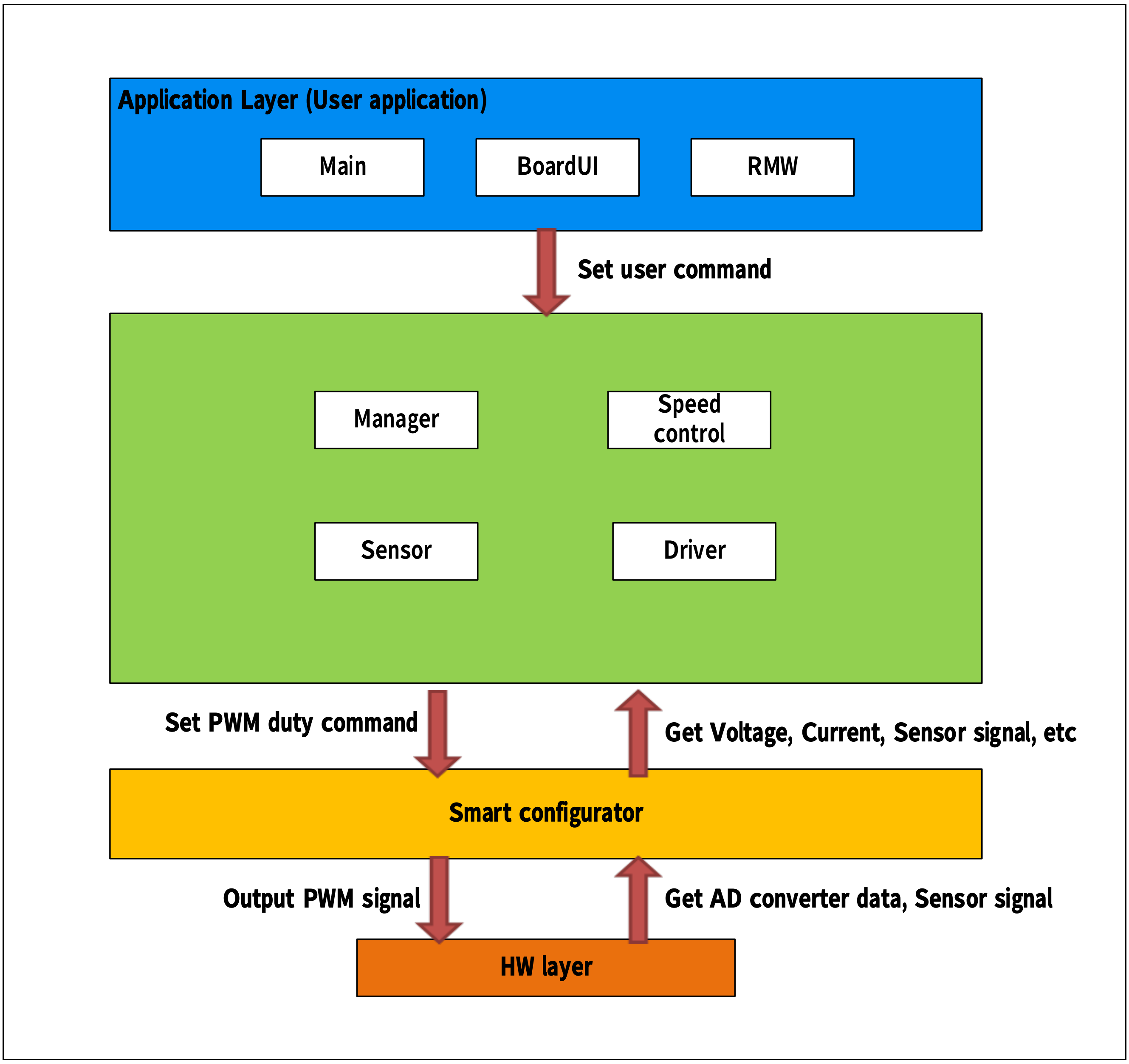


Figure 4-1 shows the overall configuration of the motor control software.

4.2.2 Configuration of the motor module

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

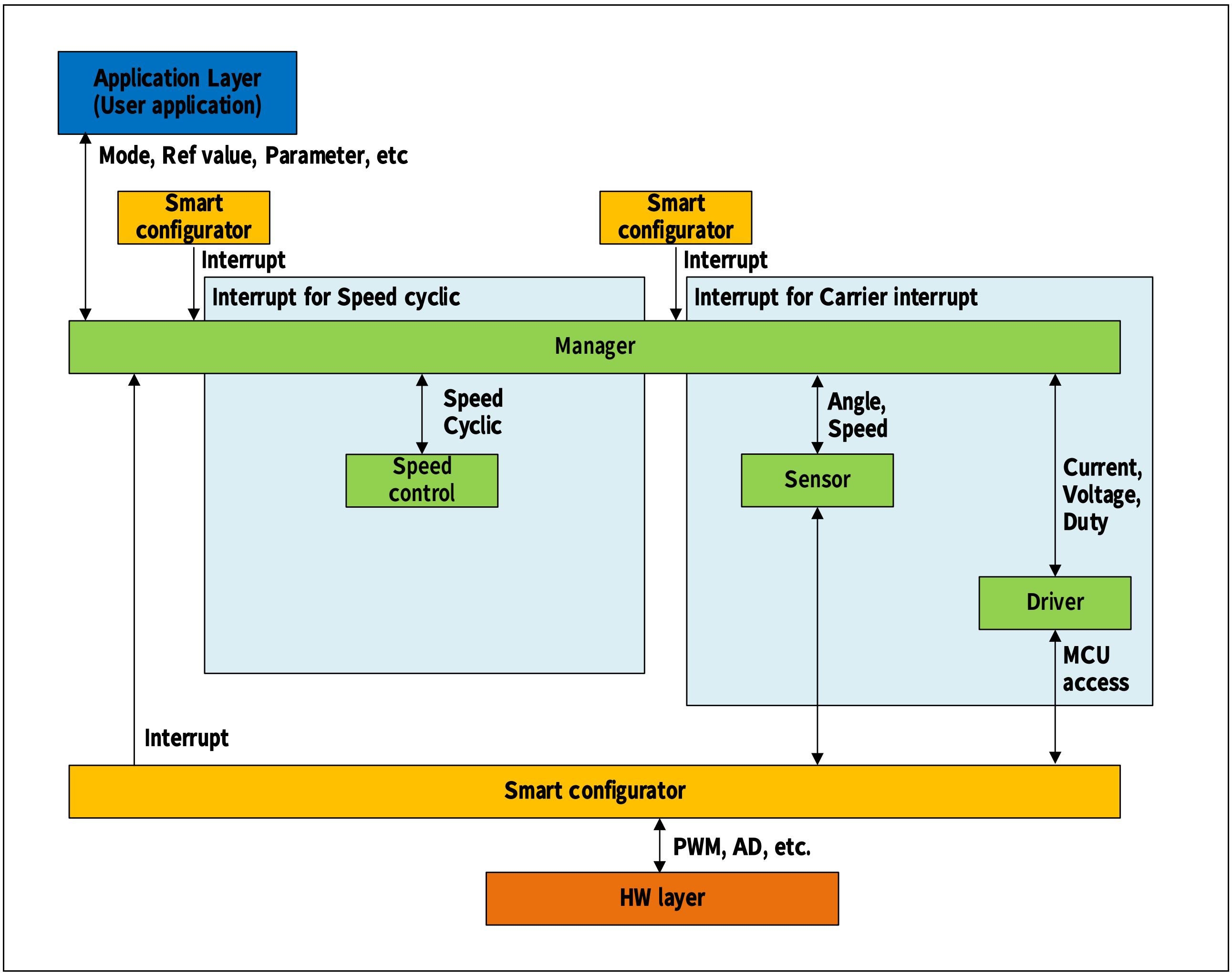


Figure 4-2 Configuration of the motor module

Table 4-2 Module summary

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
Speed control module	Module related to speed control	5.3
Sensor module	Module for acquiring position and speed information from sensor signals	5.4
Driver module	Module related to connection with Smart Configurator	5.5
Smart Configurator layer	Module related to connection with the hardware layer	5.6

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 communication
	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
motor_module	120_conduction_rx	r_motor_120_conduction_action.c	Definition of action functions
		r_motor_120_conduction_api.c/h	Definition of API functions for the manager module
		r_motor_120_conduction_manager.c/h	Definition of local functions for the manager module
		r_motor_120_conduction_protection.c/h	Definition of functions for the protection function
		r_motor_120_conduction_statemachine.c/h	Definition of functions related to state transition
	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
	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.c/h	Definition of local functions for the sensor module
	general	r_motor_filter.c/h	Definition of general-purpose filter functions
		r_motor_pi_control.c/h	Definition 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 Smart Configurator

Note: The xxx portion is replaced by an MCU name. (e.g., RX24T)

Smart Configurator can be used to generate peripheral drivers easily.

Smart Configurator saves the settings 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 RX24T as an example:

"RX24T_ESB_SPM_ 120_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 Smart Configurator.

Table 4-4 Smart Configurator folder and file configurations (RX24T)

Folder	Subfolder	2nd subfolder	File	Remarks
src	smc_gen	Config_ICU	Config_ICU.c/h	Definition of functions related to Hall interrupt controller
			Config_ICU_user.c	Definition of user functions related to Hall interrupt controller
		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_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_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_CMT1	Config_CMT1.c/h	Definition of functions related to CMT for the speed calculation
			Config_CMT1_user.c	Definition of user functions related to CMT for the speed calculation
		Config_IWDT	Config_IWDT.c/h	Definition of functions related to IWDT
			Config_IWDT_user.c	Definition of user functions related to IWDT
		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 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 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 RX24T sample>

In the 12-bit A/D converter component initialization function “R_Config_S12AD2_Create”, AN209 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, AN209 will be removed from the A/D conversion target when AN201 through AN204 are set as A/D conversion targets. Therefore, AN209 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 12-bit A/D converter component initialization function, AN201 through AN204 are removed from the A/D conversion target when AN209 is set. Therefore, AN201 through AN204 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 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 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	Obtains and sets command values for speed control.
RMW UI processing	Acquires and sets parameters (including command values for speed).

5.1.2 Module configuration diagram

Figure 5-1 shows the module configuration.

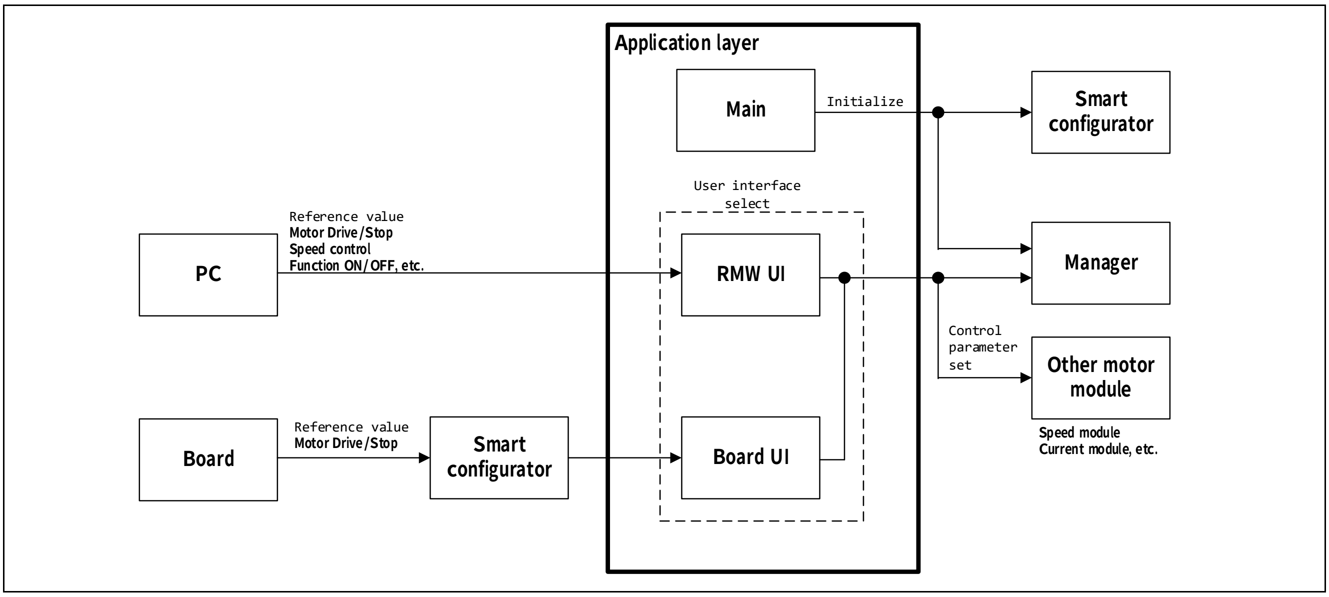


Figure 5-1 Configuration of the application layer

5.1.3 Flowcharts

(a) Main processing

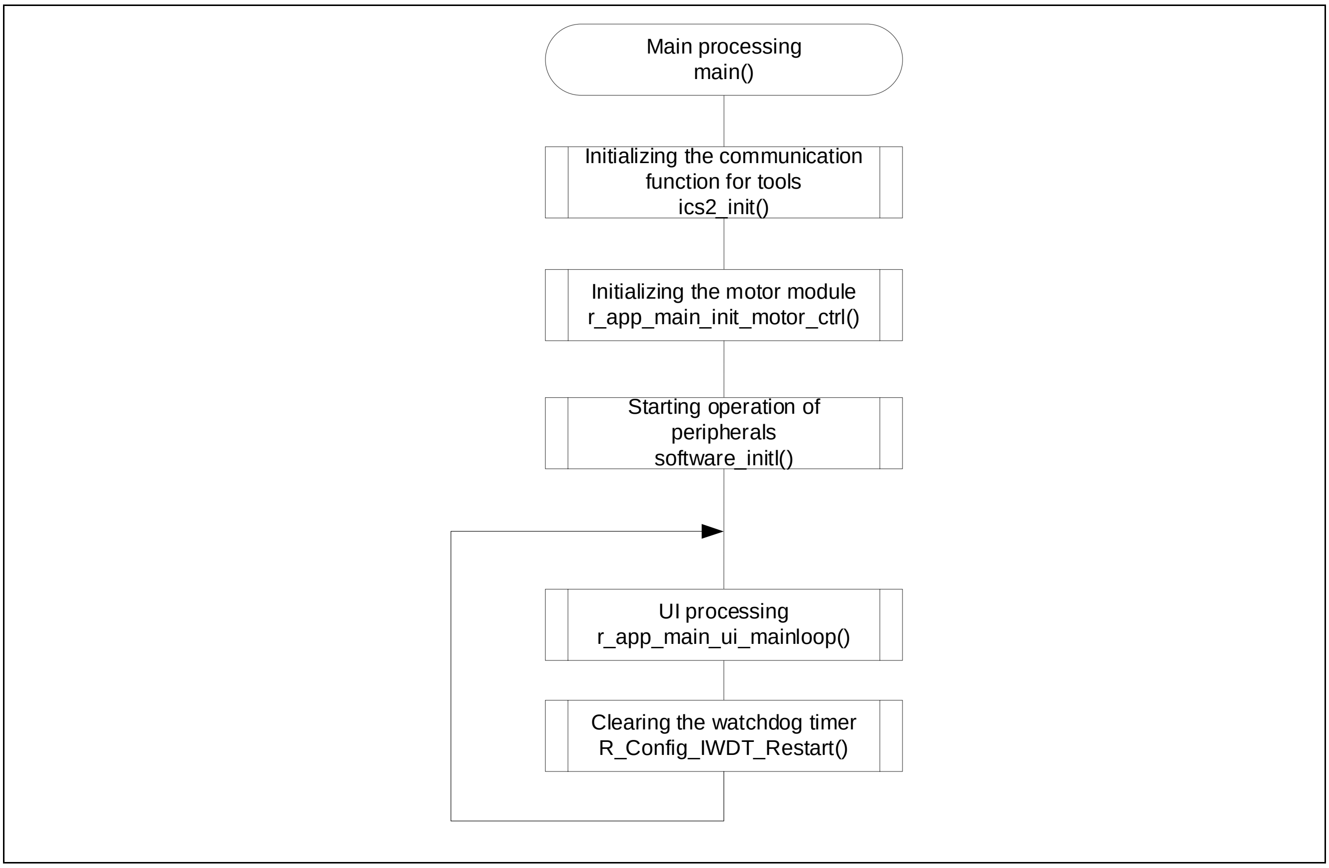


Figure 5-2 Flowchart for the main processing

(b) UI processing

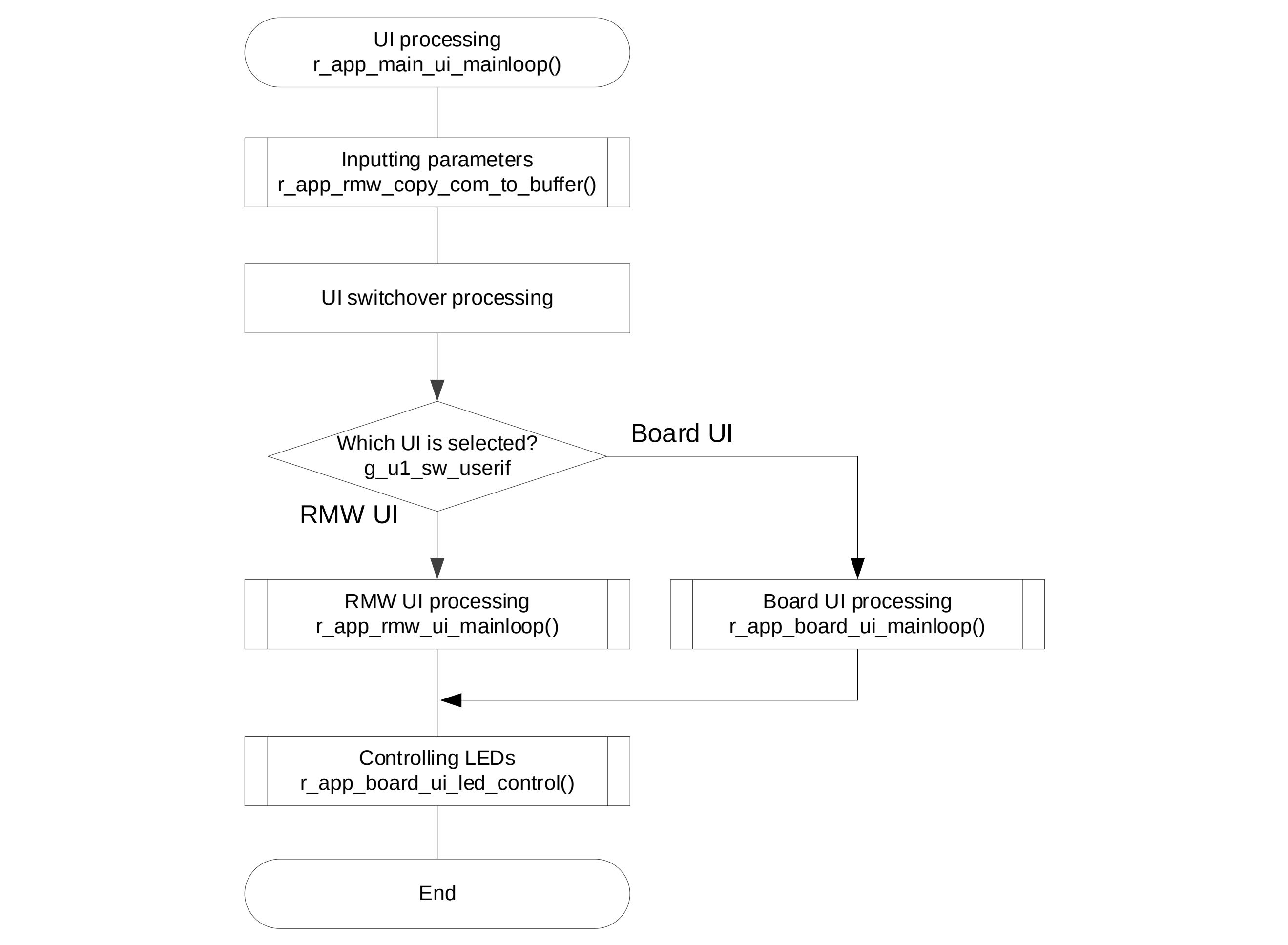


Figure 5-3 Flowchart for the UI processing

(c) Board UI processing

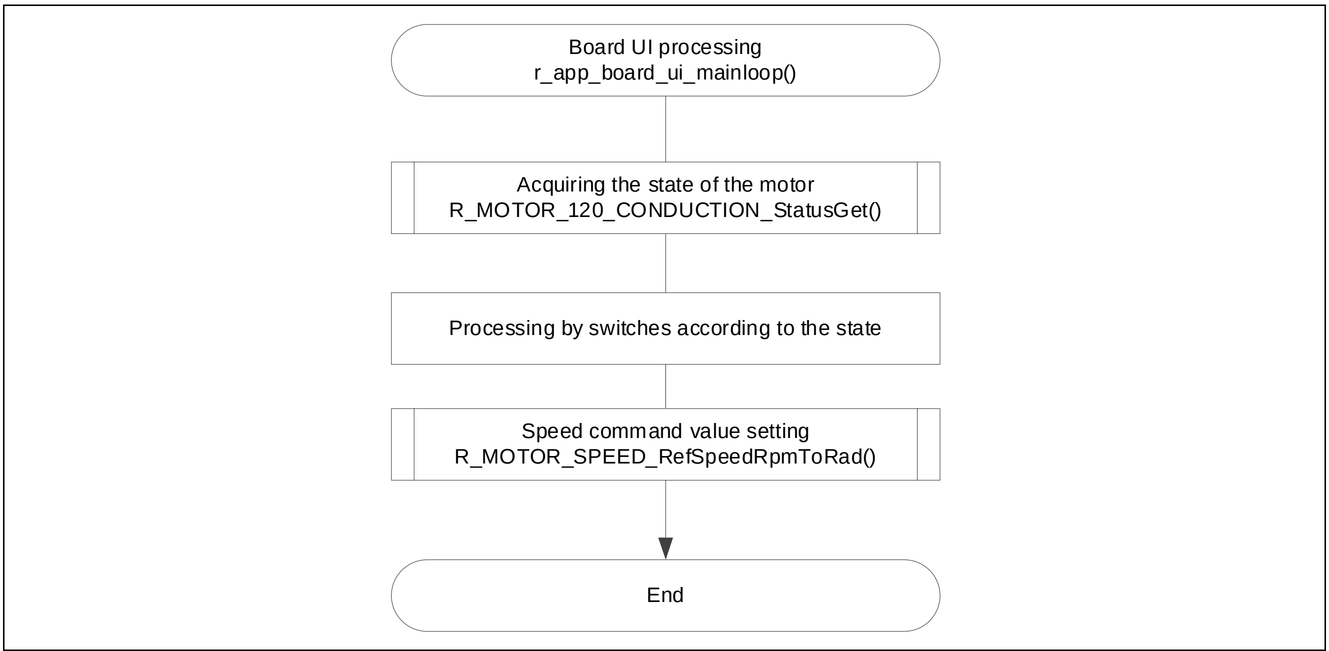


Figure 5-4 Flowchart for the Board UI processing

(d) RMW UI processing

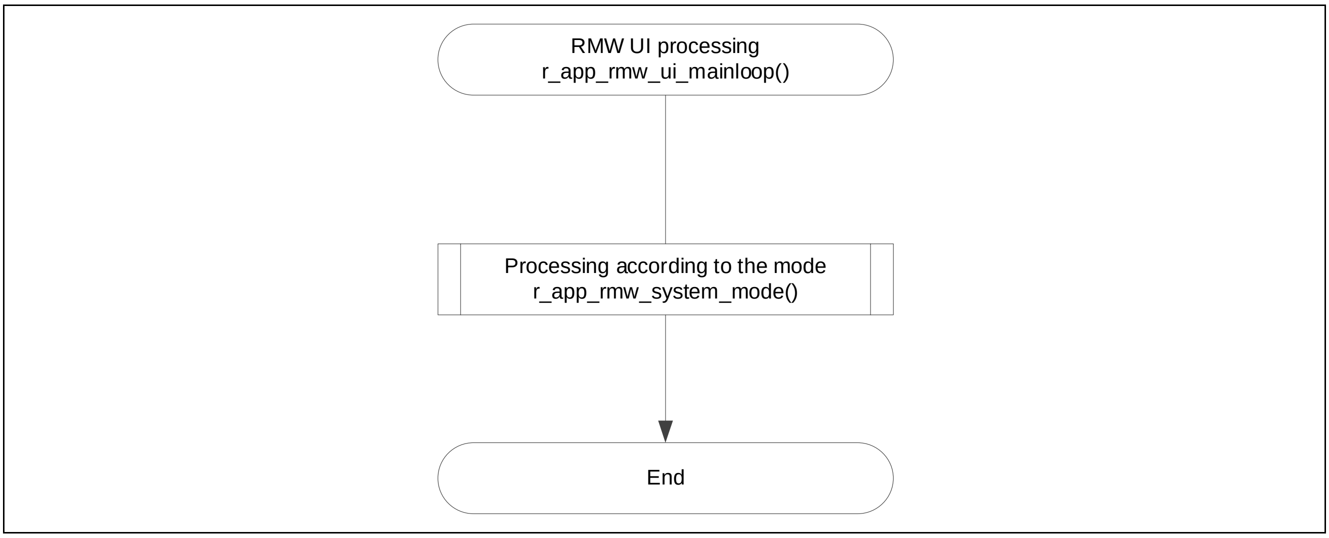


Figure 5-5 Flowchart for the RMW UI processing

5.1.4 Structure and variable information

Table 5-2 lists the variables that can be used by users in the application layer. Table 5-3 lists the members of the structure provided for updating the motor module parameters by using RMW.

Table 5-2 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_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_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	Whether to enable rewrite of variables for user entry
g_u1_enable_write	Whether to enable rewrite of variables
com_s2_ref_speed_rpm	Variable to switch the speed command value [rpm]
g_s2_ref_speed	Speed command value [rpm]
com_f4_kp_speed	Proportional term of the control gain for speed control
com_f4_ki_speed	Differential term of the control gain for speed control
com_f4_speed_lpf_k	Gain of LPF for speed control
com_u2_mtr_p	Number of pole pairs of the motor to be driven
com_f4_limit_speed_change	Acceleration setting
com_u2_offset_calc_time	Current offset measurement time setting [s]
com_f4_start_refv	Command voltage setting at startup [V]
com_u1_hall_wait_cnt	Hall interrupt time count setting
com_s2_ol_start_rpm	Speed setting at open loop startup [rpm]
com_s2_ol_mode1_change_rpm	Speed setting to transition to open loop mode 1*1 [rpm] (Sensorless only)
com_s2_ol_mode2_change_rpm	Speed setting to transition to open loop mode 2*1 [rpm] (Sensorless only)
com_f4_ol_start_refv	Command voltage setting at open loop startup [V] (Sensorless only)
com_f4_ol_mode1_rate_rpm	The speed variation setting in open loop mode 1*1 [rpm] (Sensorless only)
com_f4_ol_mode2_rate_refv	The voltage variation setting in open loop mode 2*1 [V] (Sensorless only)
com_f4_ol_mode2_rate_rpm	The speed variation setting in open loop mode 2*1 [rpm] (Sensorless only)
com_f4_ol_mode3_rate_refv	The voltage variation setting in open loop mode 3*1 [V] (Sensorless only)
com_f4_ol_mode3_max_refv	The maximum voltage setting in open loop mode 3*1 [V] (Sensorless only)
com_f4_boot_ref_v	Voltage setting in boot mode *1 [V] (Sensorless only)
com_u2_v_up_time	Voltage rising duration in boot mode [s] (Sensorless only)
com_u2_v_const_time	Constant voltage duration in boot mode [s] (Sensorless only)

Note: 1. For details, see 5.2.11, Startup sequence management.

Table 5-3 List of variables of the structure for RMW to update parameters

Structure	Variable	Description
st_rmw_param_buffer_t Structure for updating the RMW variables	s2_ref_speed	Command value [rpm]
	f4_kp_speed	Proportional term of the control gain for speed control
	f4_ki_speed	Differential term of the control gain for speed control
	f4_speed_lpf_k	Gain of LPF for speed control
	u2_mtr_p	Number of pole pairs of the motor to be driven
	f4_limit_speed_change	Acceleration setting
	u2_offset_calc_time	Current offset measurement time setting [s]
	f4_start_refv	Command voltage setting at startup [V]
	u1_hall_wait_cnt	Hall interrupt time count setting
	s2_ol_start_rpm	Speed setting at open loop startup [rpm] (Sensorless only)
	s2_ol_mode1_change_rpm	Speed setting to transition to open loop mode 1*[1] [rpm] (Sensorless only)
	s2_ol_mode2_change_rpm	Speed setting to transition to open loop mode 2*[1] [rpm] (Sensorless only)
	f4_ol_start_refv	Command voltage at open loop startup [V] (Sensorless only)
	s2_ol_start_rad	Speed setting at open loop startup [rad/s] (Sensorless only)
	f4_ol_mode1_rate_rpm	The speed variation setting in open loop mode 1*[1] [rpm] (Sensorless only)
	f4_ol_mode2_rate_refv	The voltage variation setting in open loop mode 2*[1] [V] (Sensorless only)
	f4_ol_mode2_rate_rpm	The speed variation setting in open loop mode 2*[1] [rpm] (Sensorless only)
	f4_ol_mode3_rate_refv	The voltage variation setting in open loop mode 3*[1] [V] (Sensorless only)
	f4_ol_mode3_max_refv	The maximum voltage setting in open loop mode 3*[1] [V] (Sensorless only)
	f4_boot_ref_v	Voltage setting in boot mode *[1] [V] (Sensorless only)
	u2_v_up_time	Voltage rising duration in boot mode [s] (Sensorless only)
	u2_v_const_time	Constant voltage duration in boot mode [s] (Sensorless only)

Note: 1. For details, see 5.2.11, Startup sequence management.

5.1.5 Macro definition

Table 5-4 lists macros.

Table 5-4 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_SPEED_MARGIN	50	Speed margin for VR1 [rpm]
	BOARD_VR1_SCALING_SPEED	(MOTOR_CFG_MAX_SPEED_RPM + 50.0f) / 2048	Speed scaling coefficient for VR1
r_app_rmw.h	ICS_DECIMATION	3	RMW watchpoint skip count
	ICS_INT_LEVEL	6	RMW interrupt priority
	ICS_BRR	4	Communication baud rate for RMW
	ICS_INT_MODE	1	Communication mode selection for RMW
	ICS_SCI_CH_SELECT	0x60	SCI channel to be used

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

Table 5-5 List of macro difinitions

Macro name	RX13T	RX23T	RX24T, RX24U
ICS_DECIMATION	3	3	3
ICS_INT_LEVEL	6	6	6
ICS_BRR	3	4	4
ICS_INT_MODE	1	1	1

5.1.6 Adjustment and configuration of parameters

For the variables listed in Table 5-2, perform adjustment and configuration from RMW. For details about how to use RMW, see 3, Quick start guide and the Renesas Motor Workbench V.2.00 User's Guide (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 5-6 lists the functions of the manager module.

Table 5-6 List of manager module functions

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

5.2.2 Module configuration diagram

Figure 5-6 shows the module configuration.

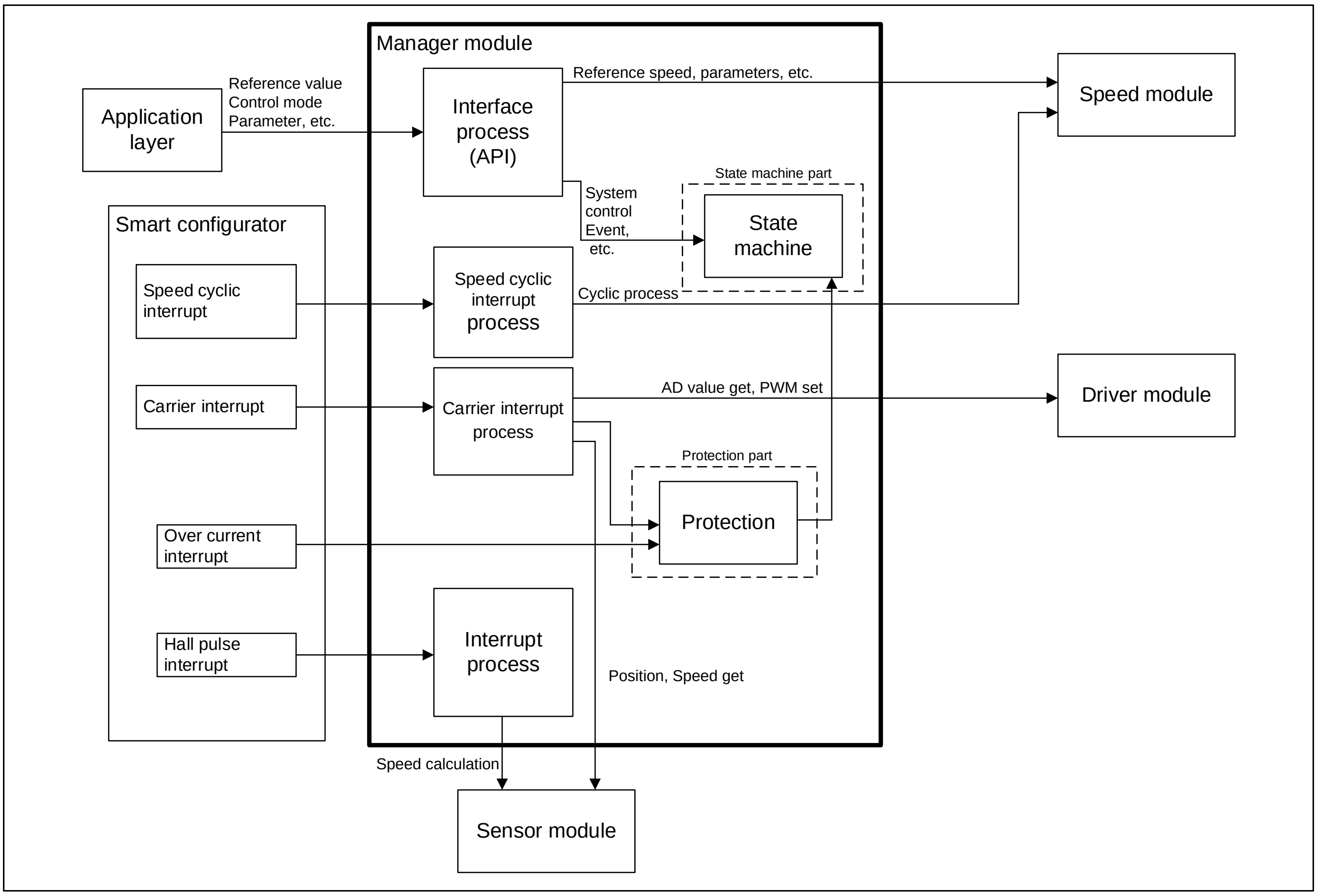


Figure 5-6 Manager module configuration diagram

5.2.3 Mode management

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

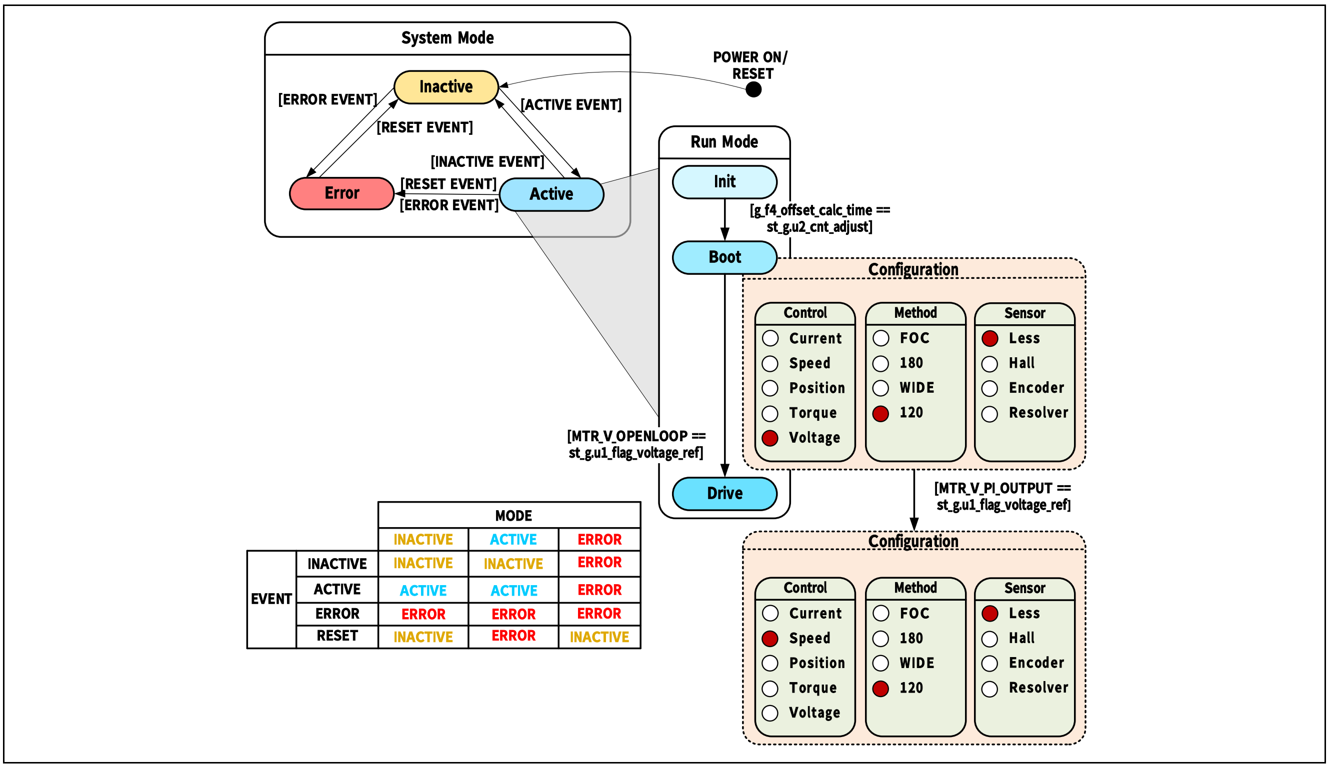


Figure 5-7 State transition diagram for the encoder-based vector control software

(1) System modes

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

(2) Run modes

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

(3) Events

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

Table 5-7 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 5-8.

- Overcurrent error
  Overcurrent errors can be detected on the hardware and in the software.
  A high-impedance output is provided to the PWM output pin in response to an emergency stop signal (overcurrent detection) from the hardware.
  This function monitors U-, V-, and W-phases at the overcurrent monitoring interval. When this function detects an overcurrent (the status in which the current is above the overcurrent limit value), it brings the program to an emergency stop (software detection).
  The overcurrent limit value is automatically calculated from the rated current of the motor (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 5-8 Values specified for the system protection function settings

Overcurrent error	Overcurrent limit value [A]	0.89
	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]	3000
	Monitoring interval [μs]	Current control interval *1

Note: 1. See Table 4-1 Basic specifications of 120-degree conducting control software.

5.2.5 Flowcharts

The manager module performs processing in response to the occurrences of interrupts that are set in 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) Carrier wave interrupt processing

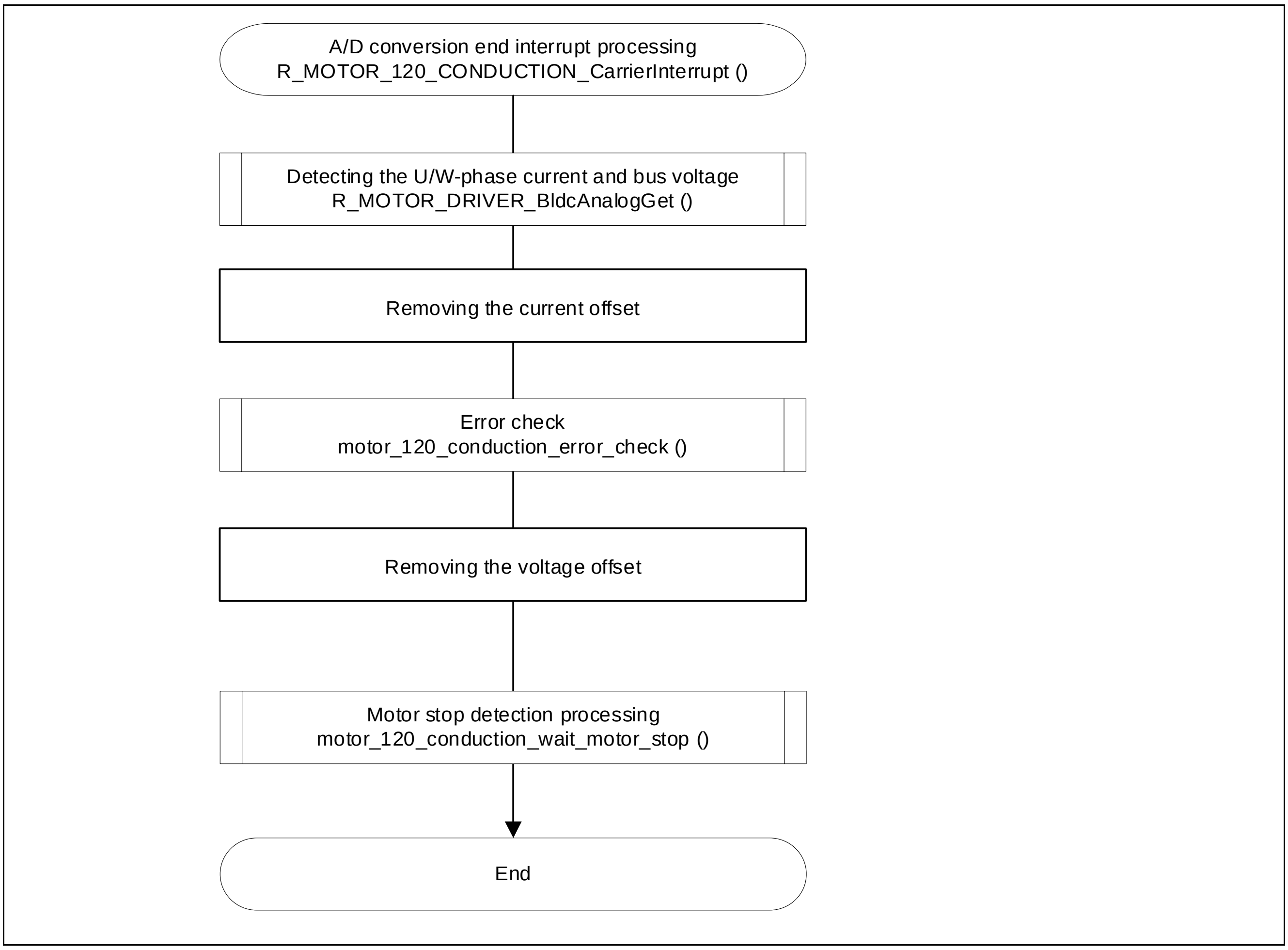


Figure 5-8 Carrier wave interrupt processing flowchart(in a configuration with a Hall sensor)

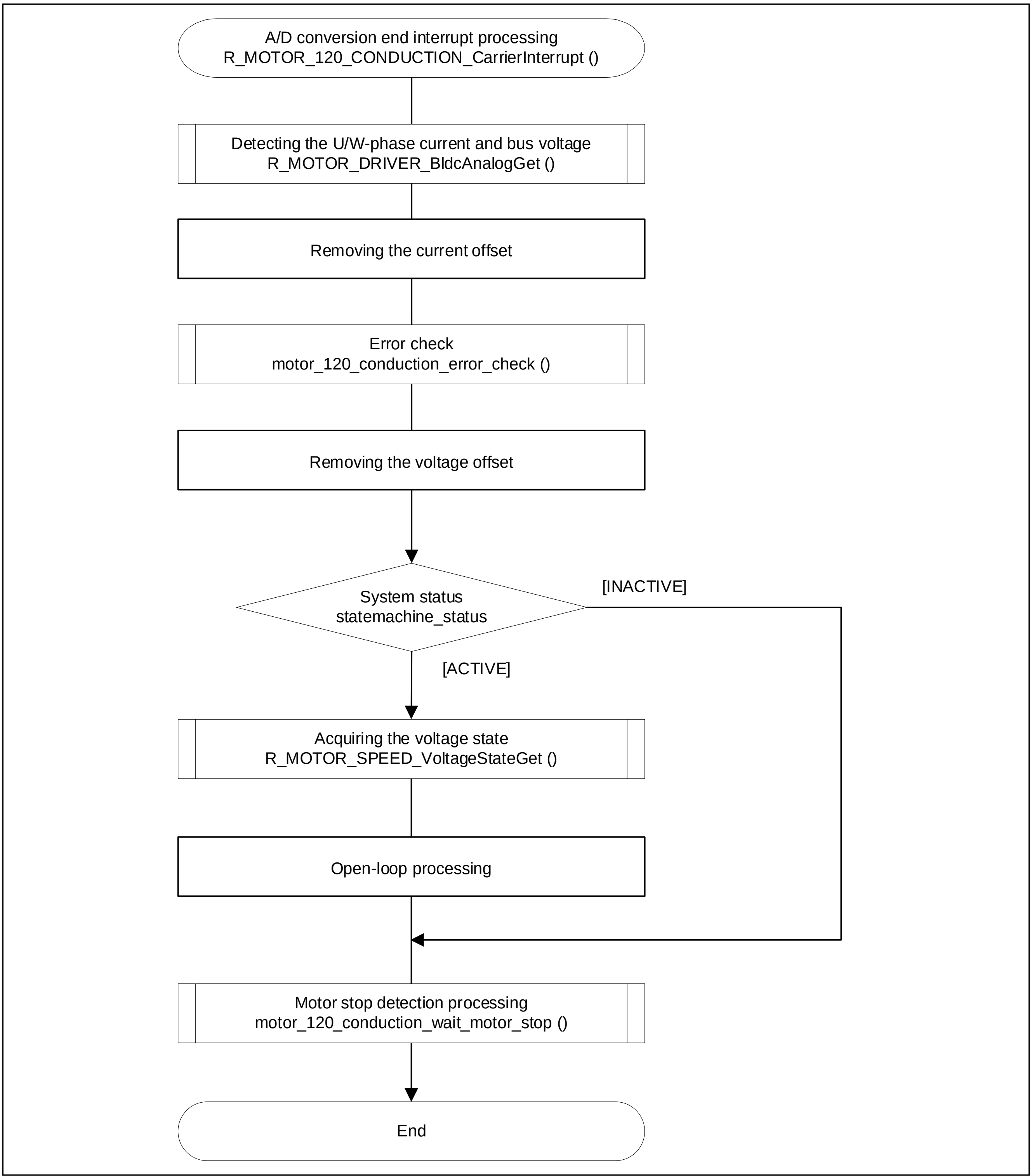


Figure 5-9 Carrier wave interrupt processing flowchart (in a sensorless configuration)

(b) Interrupt processing for speed control

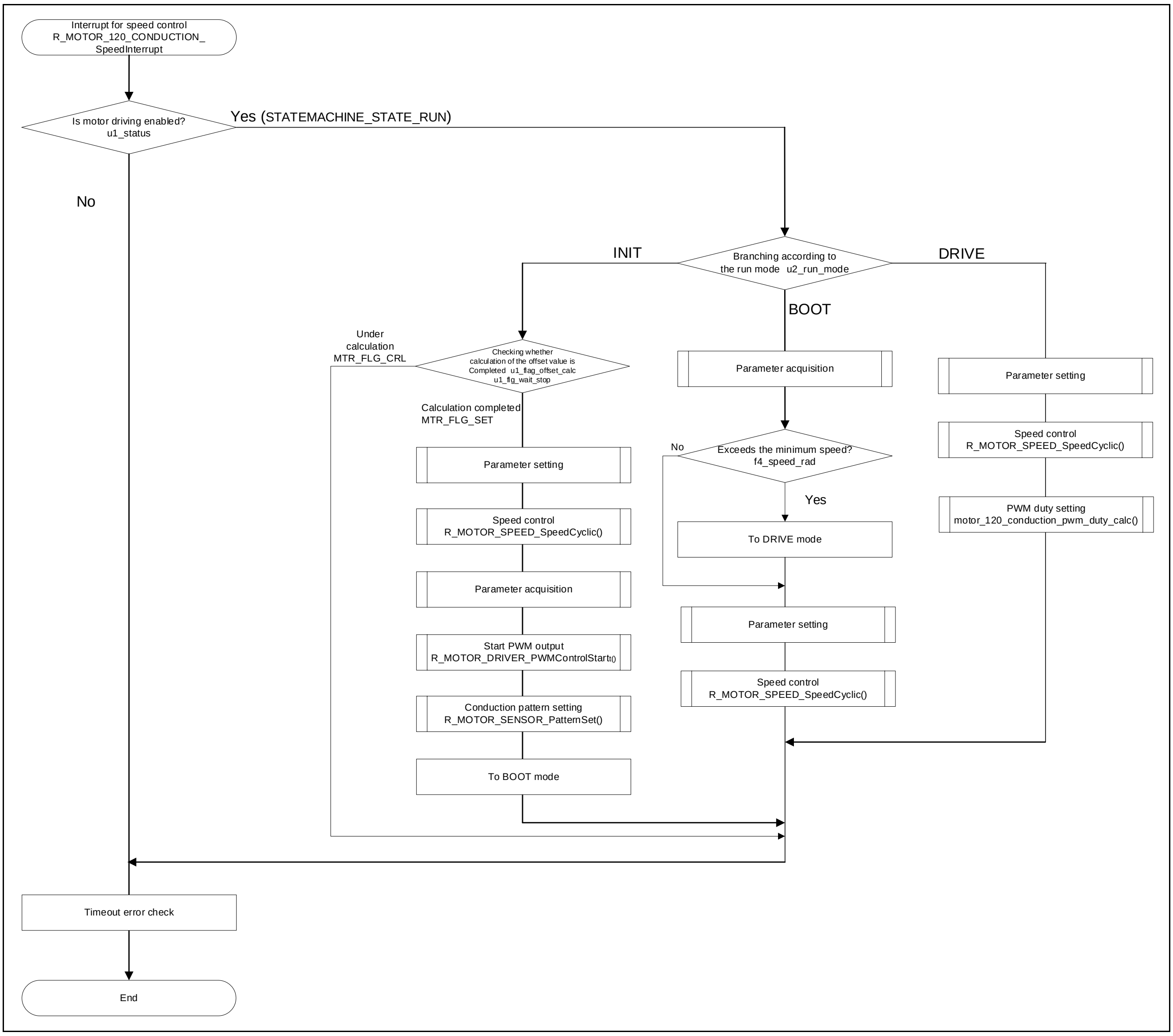


Figure 5-10 Interrupt processing flowchart for speed control (in a configuration with a Hall sensor)

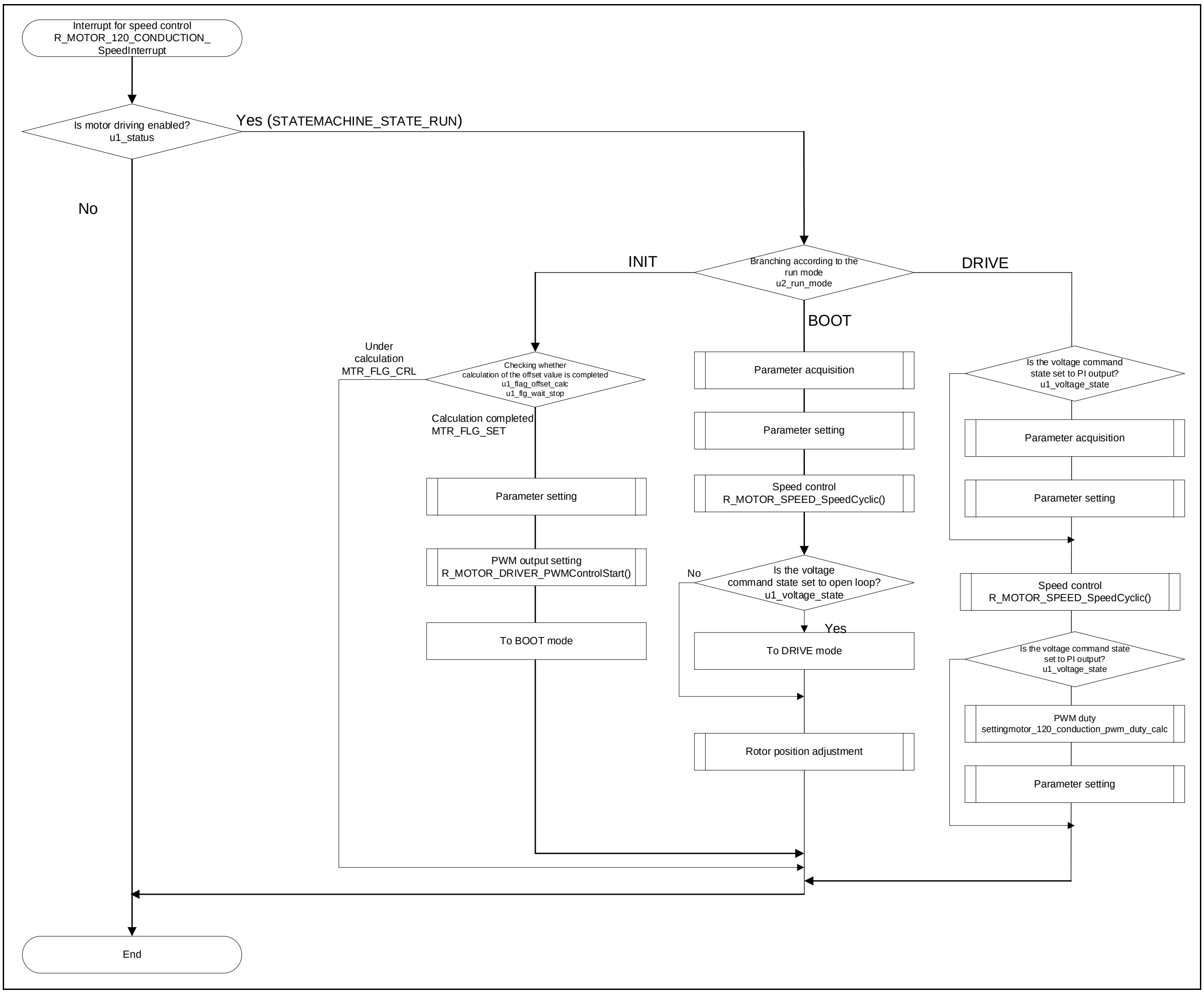


Figure 5-11 Interrupt processing flowchart for speed control (in a sensorless configuration)

(c) Overcurrent detection interrupt processing

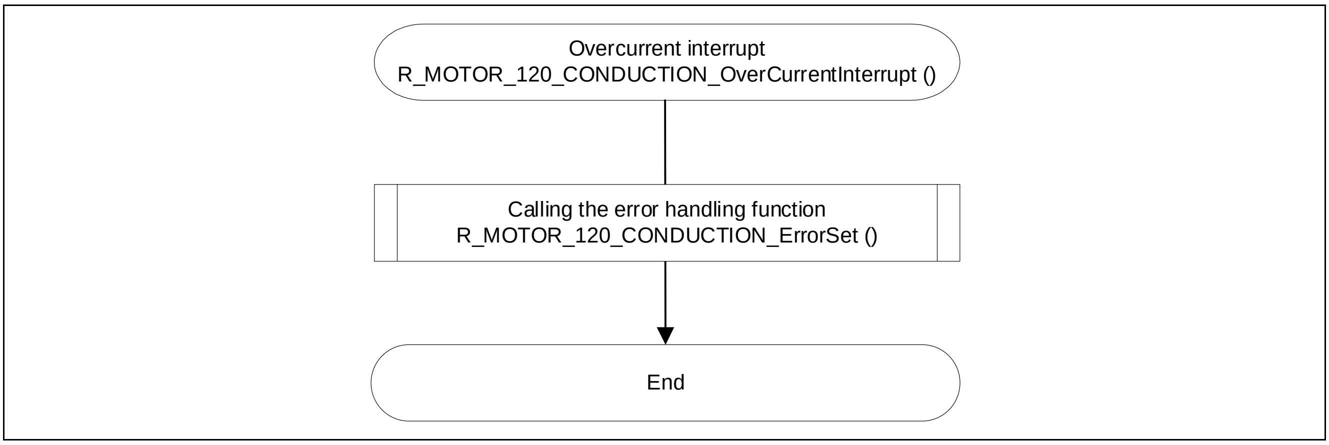


Figure 5-12 Overcurrent detection interrupt processing flowchart

(d) Interrupt processing for the Hall signal

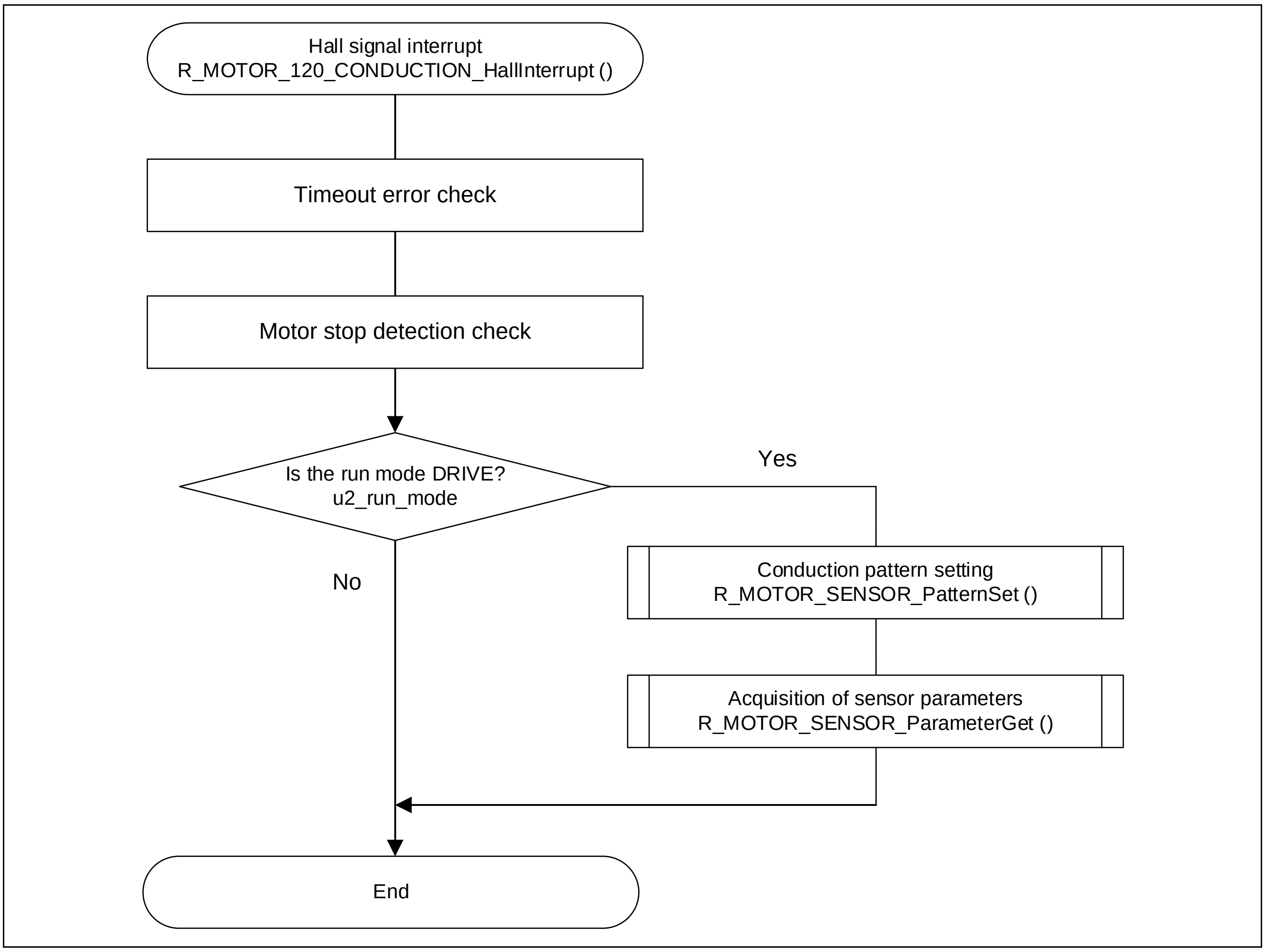


Figure 5-13 Hall edge interrupt processing flowchart

5.2.6 API

Table 5-9 lists the manager module API functions.

Table 5-9 List of API functions

API	Description
R_MOTOR_120_CONDUCTION_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_120_CONDUCTION_Close	Places the modules, including the manager module, in a reset state.
R_MOTOR_120_CONDUCTION_Reset	Initializes the modules, including the manager module.
R_MOTOR_120_CONDUCTION_ParameterUpdate	Updates the control parameter settings of this module. This function also updates the control parameters for the related modules.
R_MOTOR_120_CONDUCTION_MotorStart	Places the motor in the running state.
R_MOTOR_120_CONDUCTION_MotorStop	Places the motor in the stopped state.
R_MOTOR_120_CONDUCTION_MotorReset	Releases the system from the error state.
R_MOTOR_120_CONDUCTION_ErrorSet	Places the system in an error state.
R_MOTOR_120_CONDUCTION_SpeedSet	Sets the speed command value. This function is enabled when speed control is being performed.
R_MOTOR_120_CONDUCTION_StatusGet	Acquires the status from the state machine.
R_MOTOR_120_CONDUCTION_ErrorStatusGet	Acquires the error state.
R_MOTOR_120_CONDUCTION_SpeedInterrupt	Performs interrupt processing for speed control.
R_MOTOR_120_CONDUCTION_CarrierInterrupt	Performs interrupt processing for carrier wave interrupt processing.
R_MOTOR_120_CONDUCTION_OverCurrentInterrupt	Performs interrupt processing when an overcurrent occurs.
R_MOTOR_120_CONDUCTION_HallInterrupt	Performs interrupt processing for Hall sensor signals.

5.2.7 Configurations

Table 5-10 lists the configurations for the manager module.

Table 5-10 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_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_CFG_MAX_SPEED_RPM	Maximum speed [rpm]
	MOTOR_CFG_MIN_SPEED_RPM	Minimum speed [rpm]
	MOTOR_CFG_MARGIN_SPEED	Margin around the threshold between sensorless switching [rpm]
	MOTOR_CFG_MARGIN_MIN_SPEED	Minimum speed considering the margin [rpm]

Table 5-11 List of initial values for configurations

Macro name	RX13T	RX23T	RX24T, RX24U
MOTOR_MCU_CFG_PWM_TIMER_FREQ	32	40	80
MOTOR_MCU_CFG_CARRIER_FREQ	16	20	
MOTOR_MCU_CFG_AD_FREQ	32	40	40
MOTOR_MCU_CFG_AD_SAMPLING_CYCLE	47	47	47
MOTOR_MCU_CFG_AD12BIT_DATA	4095.0f		
MOTOR_MCU_CFG_ADC_OFFSET	0x7FF		
MOTOR_CFG_MAX_SPEED_RPM	2650		
MOTOR_CFG_MIN_SPEED_RPM	With Hall sensor: 550 Sensorless: 1000		
MOTOR_CFG_MARGIN_SPEED	50		
MOTOR_CFG_MARGIN_MIN_SPEED	MOTOR_CFG_MIN_SPEED_RPM - MOTOR_CFG_MARGIN_SPEED		

5.2.8 Structure and variable information

Table 5-12 lists the structures and variables for the manager module. For the manager module, the structure for the manager module (g_st_120_conduction) is defined by securing an instance of the module from the API.

Table 5-12 List of structures and variables

Structure	Variable	Description
st_120_conduction_control_t Structure for the manager module	u1_mode_system	System mode
	u2_run_mode	Run mode 0: Initialization 1: Startup preparation 2: Driving the motor
	u2_error_status	Error status
	f4_vdc_ad	Bus voltage [V]
	f4_v_ref	Voltage command value [V]
	f4_start_refv	Voltage command value at startup [V]
	f4_vu_ad	U-phase voltage [V]
	f4_vv_ad	V-phase voltage [V]
	f4_vw_ad	W-phase voltage [V]
	f4_vn_ad	Center point voltage [V]
	f4_offset_vu	U-phase voltage offset [V]
	f4_offset_vv	V-phase voltage offset [V]
	f4_offset_vw	W-phase voltage offset [V]
	f4_offset_off_vu	U-phase voltage when the inverter is stopped [V]
	f4_offset_off_vv	V-phase voltage when the inverter is stopped [V]
	f4_offset_off_vw	W-phase voltage when the inverter is stopped [V]
	f4_sum_vu_ad	U-phase total voltage value [V]
	f4_sum_vv_ad	V-phase total voltage value [V]
	f4_sum_vw_ad	W-phase total voltage value [V]
	u1_flag_draw_in	Flag for initial position setting
	f4_iu_ad	U-phase current [A]
	f4_iv_ad	V-phase current [A]
	f4_iw_ad	W-phase current [A]
	f4_offset_iu	U-phase current_offset [A]
	f4_offset_iv	V-phase current offset [A]
	f4_offset_iw	W-phase current offset [A]
	f4_sum_iu_ad	U-phase total current value [V]
	f4_sum_iv_ad	V-phase total current value [V]
	f4_sum_iw_ad	W-phase total current value [V]
	f4_inv_offset_calc	Inverse of the offset calculation time
	u2_offset_calc_time	Offset calculation time [s]
	u1_flag_offset_calc	Flag for the offset calculation
	u2_cnt_adjust	Offset time counter
	u1_direction	Rotation direction

Structure	Variable	Description
st_120_conduction_control_t Structure for the manager module	u2_cnt_timeout	Timeout count value
	u2_hall_timer_cnt	Timer count value for Hall
	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]
	u2_hall_timeout_cnt_limit	Timeout count limit value for Hall
	u2_pwm_duty	PWM duty value
	f4_speed_rad	Speed [rad/s]
	f4_rpm_rad	Variable for unit conversion (rpm to rad/s)
	u1_flg_wait_stop	Flag for rotation stop detection
	u2_cnt_wait_stop	Count value for rotation stop detection
	u1_hall_wait_cnt	Count value for Hall stop detection
	st_speed_output	Structure for speed control module output
	st_sensor_output	Structure for sensor module output
	st_motor	Motor parameter structure
	*p_st_driver	Pointer to the structure for the driver module
	*p_st_speed	Pointer to the structure for the speed module
	*p_st_sensor	Pointer to the structure for the sensor module
st_120_conduction_cfg Structure for setting the manager module control parameters	u1_hall_wait_cnt	Count value for Hall stop detection
	u2_offset_calc_time	Offset calculation time [s]
	u2_mtr_p	Number of pole pairs of the motor
	u1_direction	Rotation direction
	f4_rpm_rad	Variable for unit conversion (rpm to rad/s)

5.2.9 Macro definition

Table 5-13 lists the macros for the manager module.

Table 5-13 List of macros

File name	Macro name	Defined value	Remarks
r_motor_120_conduction_api.h	MOTOR_MODE_INIT	0x00	Initialization mode
	MOTOR_MODE_BOOT	0x01	Startup preparation mode
	MOTOR_MODE_DRIVE	0x02	Motor drive mode
	MOTOR_MODE_ANALYSIS	0x03	Analysis mode (Not available)
	MOTOR_MODE_TUNE	0x04	Automatic adjustment mode (Not available)
	MOTOR_ERROR_NONE	0x0000	No errors detected.
	MOTOR_ERROR_OVER_CURRENT_HW	0x0001	An error status. A hardware overcurrent error has occurred.
	MOTOR_ERROR_OVER_VOLTAGE	0x0002	An error status. An overvoltage error has occurred.
	MOTOR_ERROR_OVER_SPEED	0x0004	An error status. An overspeed error has occurred.
	MOTOR_ERROR_HALL_TIMEOUT	0x0008	An error status. Interrupt detection timeout error has occurred for a Hall configuration.
	MOTOR_ERROR_BEMF_TIMEOUT	0x0010	An error status. Counter-electromotive force detection timeout error has occurred.
	MOTOR_ERROR_HALL_PATTERN	0x0020	An error status. A conduction pattern error has occurred for a Hall configuration.
	MOTOR_ERROR_BEMF_PATTERN	0x0040	An error status. A conduction pattern error has occurred for a sensorless configuration.
	MOTOR_ERROR_LOW_VOLTAGE	0x0080	An error status. A low-voltage error has occurred.
	MOTOR_ERROR_OVER_CURRENT_SW	0x0100	An error status. A software overcurrent error has occurred.
	MOTOR_ERROR_UNKNOWN	0xffff	An error status. An error whose error code is unknown has occurred.
	MOTOR_SENSOR_LESS	0x01	Drives a motor in a sensorless configuration.
	MOTOR_SENSOR_HALL	0x02	Drives a motor by using a Hall sensor.
	MOTOR_SENSOR_ENCD	0x04	Drives a motor by using an encoder. (Not available)
	MOTOR_SENSOR_RESO	0x08	Drives a motor by using a resolver sensor. (Not available)
	MOTOR_METHOD_FOC	0x00	Drives a motor using vector control. (Not available)
	MOTOR_METHOD_180	0x01	Drives a motor with 180-degree conducting control. (Not available)
	MOTOR_METHOD_WIDE	0x02	Drives a motor using wide-angle control. (Not available)
	MOTOR_METHOD_120	0x03	Drives a motor with 120-degree conducting control.
	MOTOR_CONTROL_CURRENT	0x01	Performs current control. (Not available)

File name	Macro name	Defined value	Remarks
r_motor_120_conduction_api.h	MOTOR_CONTROL_SPEED	0x02	Performs speed control.
	MOTOR_CONTROL_POSITION	0x04	Performs position control. (Not available)
	MOTOR_CONTROL_TORQUE	0x08	Performs torque control. (Not available)
	MOTOR_CONTROL_VOLTAGE	0x10	Performs voltage control. (Not available)
r_motor_120_conduction_statemachine.h	STATEMACHINE_STATE_STOP	0x00	Stopped state
	STATEMACHINE_STATE_RUN	0x01	Running state
	STATEMACHINE_STATE_ERROR	0x02	Error state
	STATEMACHINE_STATE_SIZE	3	State size
	STATEMACHINE_EVENT_STOP	0x00	Stoppage event
	STATEMACHINE_EVENT_RUN	0x01	Run event
	STATEMACHINE_EVENT_ERROR	0x02	Error event
	STATEMACHINE_EVENT_RESET	0x03	Reset event
	STATEMACHINE_EVENT_SIZE	4	Event size

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 5-14 shows the values set in the sample program.

Table 5-14 Motor and inverter parameter settings

File name	Macro name	Set value	Description
r_motor_inverter_cfg.h	INVERTER_CFG_DEADTIME	2.0f	Dead time [μs]
	INVERTER_CFG_CURRENT_RANGE	25.0f	Scaling for current
	INVERTER_CFG_VDC_RANGE	111.0f	Scaling for voltage
	INVERTER_CFG_INPUT_V	24.0f	Input voltage [V]
	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_ADC_REF_VOLTAGE	5.0f	Analog power supply voltage for the MCU [V]
r_motor_targetmotor_cfg.h	MOTOR_CFG_POLE_PAIRS	2	Number of pole pairs
	MOTOR_CFG_MAGNETIC_FLUX	0.02159f	Magnetic flux [wb]
	MOTOR_CFG_RESISTANCE	6.447f	Resistance [ohm]
	MOTOR_CFG_D_INDUCTANCE	0.0045f	d-axis inductance [H]
	MOTOR_CFG_Q_INDUCTANCE	0.0045f	q-axis inductance [H]
	MOTOR_CFG_NOMINAL_CURRENT_RMS	0.42f	Rated current [A]

5.2.11 Startup sequence management

(a) Hall sensor

In the 120-degree conducting control using a Hall sensor, the conduction pattern at startup is uniquely determined because the rotor position can be identified by Hall sensor signals.

However, to control the speed, it is necessary to measure the time data for at least the first 2π as shown in 5.4.9. Therefore, the sample program uses an open-loop method with a constant voltage for startup, waits for a condition where time data can be acquired, and then transitions to speed control.

Figure 5-14 shows how the sample program starts the motor. “MTR_MODE_BOOT” starts up the motor in open-loop at a constant voltage given by g_st_speed.f4_v_ref. The condition for transition to “MTR_MODE_DRIVE” is that the RPM measured at that point reaches the specified minimum RPM (550rpm).

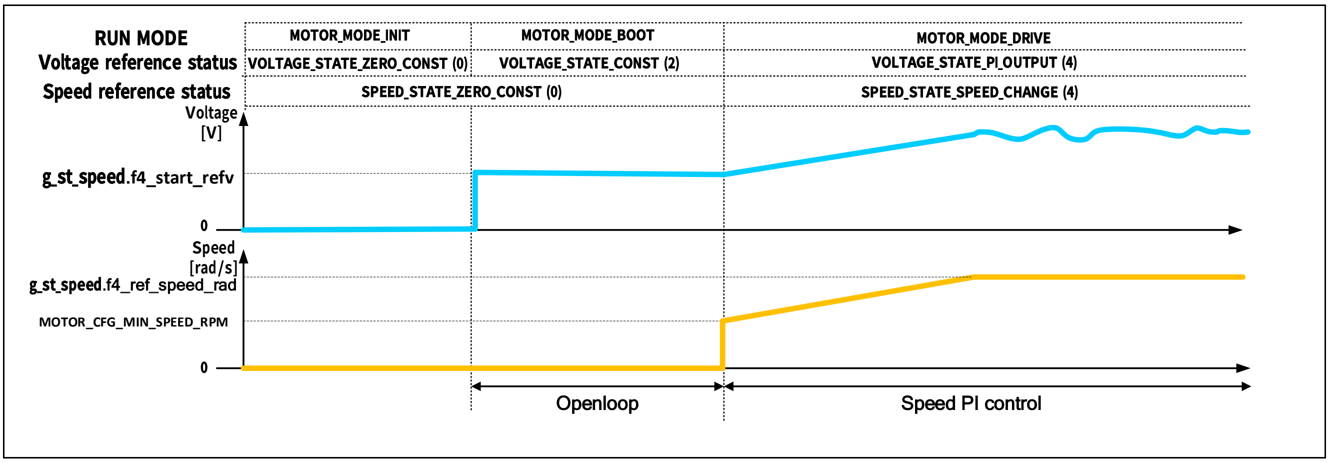


Figure 5-14 Example of startup based on 120-degree conducting control using a Hall sensor

(b) Sensorless

In the sensorless 120-degree conducting control, the induced voltage generated from the magnetic flux of the permanent magnet (rotor) is used to estimate the magnetic pole position every 60 degrees. However, because the induced voltage is generated by rotation, the position of the magnetic pole cannot be estimated at startup. Sufficient rotation speed is required to estimate the magnetic pole position.

Therefore, one way to start a motor is to forcibly change the conduction pattern regardless of the position of the permanent magnet to generate a rotating magnetic field and make the motor rotate at a synchronous speed.

Figure 5-15 shows how the sample program starts the motor. “MOTOR_MODE_BOOT” pulls in the magnetic pole position to prevent overcurrent during startup.

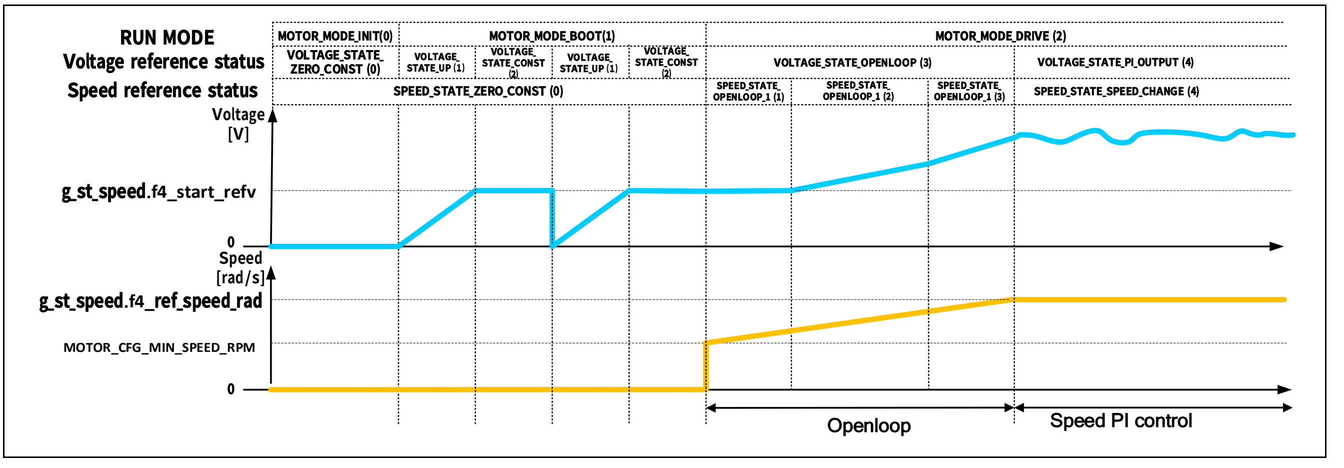


Figure 5-15 Example of startup based on sensorless 120-degree conducting control

5.2.12 Voltage control by PWM

The output voltage is PWM-controlled. The PWM control is a control method that continually adjust the average voltage by varying the duty cycle of the pulses as shown in Figure 5-16.

$$\text{Duty} = \frac{T_{ON}}{T_{ON} + T_{OFF}} \times 100[\%]$$

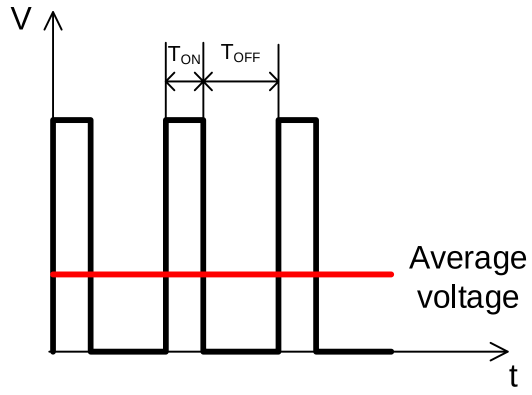


Figure 5-16 PWM control

Here, the modulation rate m is defined as follows:

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

m: *Modulation rate* V: *Command value voltage* E: *Inverter bus voltage*

This modulation rate is reflected in the register setting that determines the PWM duty cycle.

In the sample program, chopping control is adopted at the first 60 degrees to control the output voltage and speed. An example of motor control signal output waveform during noncomplementary first 60-degrees chopping is shown in Figure 5-17. An example of motor control signal output waveform during complementary first 60-degrees chopping is shown in Figure 5-18.

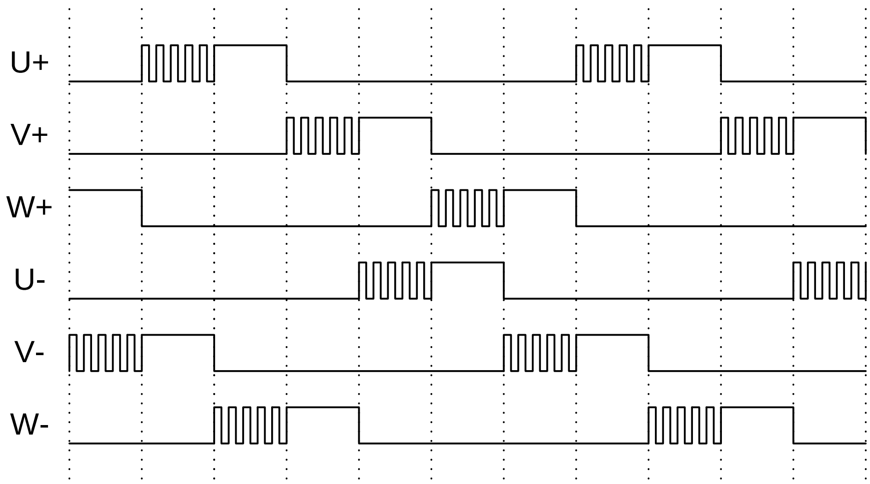


Figure 5-17 Noncomplementary first 60-degrees chopping

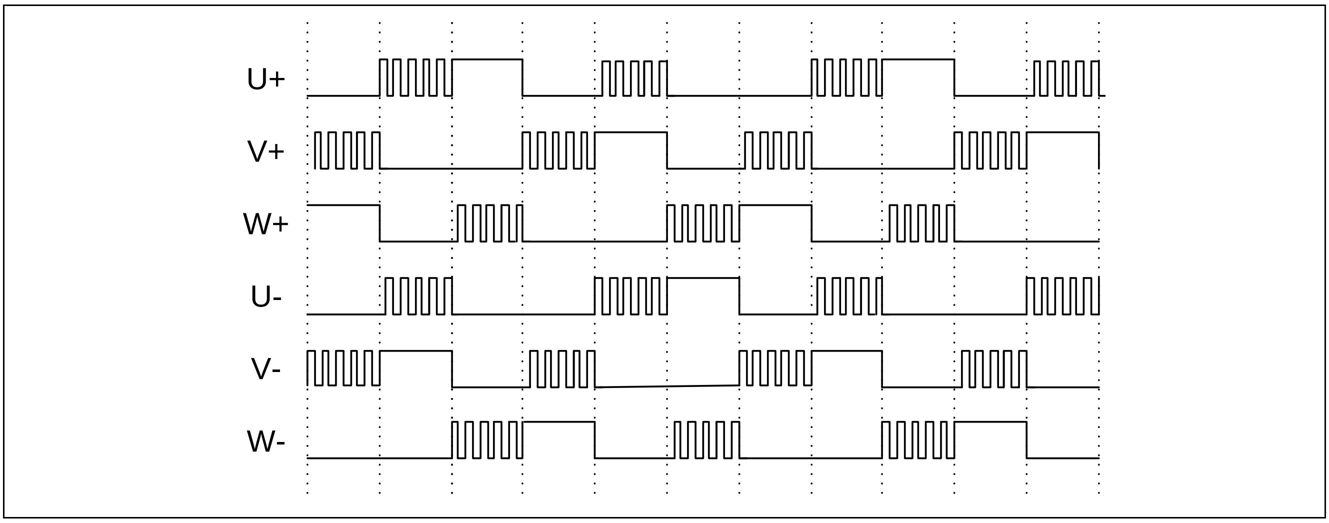


Figure 5-18 Complementary first 60-degrees chopping

5.3 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 voltage command value accordingly.

5.3.1 Functions

Table 5-15 lists the functions of the speed control module.

Table 5-15 List of functions of the speed control module

Function	Description
Speed control	Calculates and outputs a voltage 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.

5.3.2 Module configuration diagram

Figure 5-19 shows the module configuration of the speed control module.

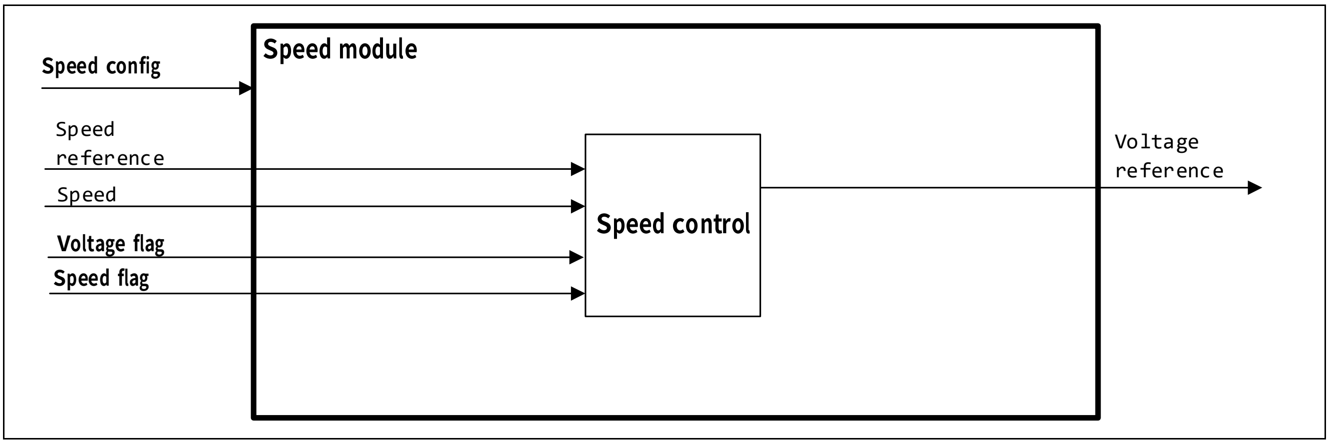


Figure 5-19 Speed control module configuration diagram

5.3.3 Flowchart

Figure 5-20 shows the flowchart for speed control.

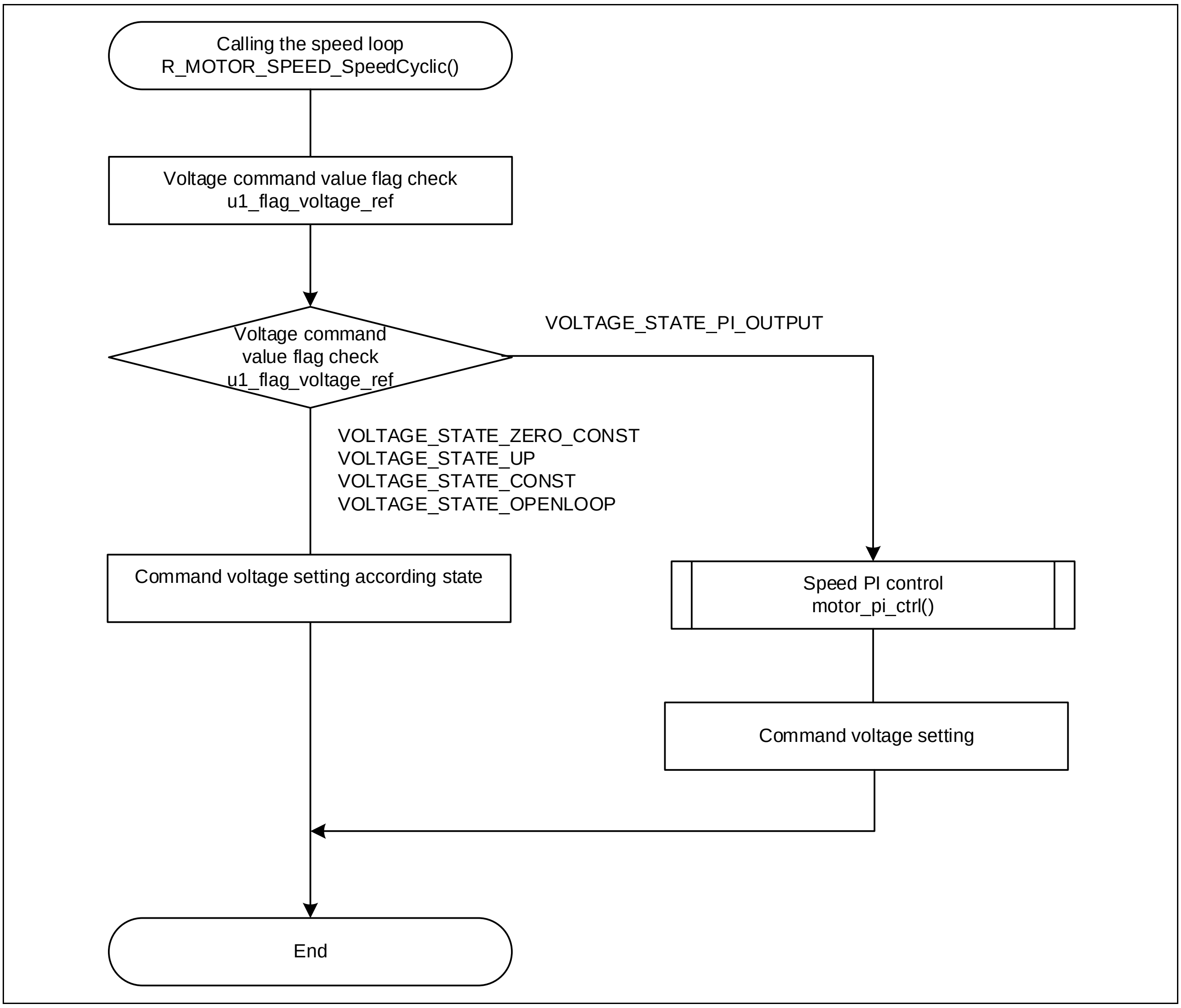


Figure 5-20 Flowchart for speed control

5.3.4 API functions

Table 5-16 lists the API functions of the speed control module.

Table 5-16 List of API functions

API	Description
R_MOTOR_SPEED_Open	Generates an instance of the speed control module.
R_MOTOR_SPEED_Close	Places the module in a reset state.
R_MOTOR_SPEED_Reset	Initializes the module.
R_MOTOR_SPEED_Run	Activates the module.
R_MOTOR_SPEED_ParameterSet	Inputs the variable information that is used for speed control.
R_MOTOR_SPEED_ParameterGet	Acquires the speed control results that are output.
R_MOTOR_SPEED_ParameterUpdate	Updates the control parameters of the module.
R_MOTOR_SPEED_SpeedCyclic	Performs speed control.
R_MOTOR_SPEED_RefSpeedSet	Sets the speed command value.
R_MOTOR_SPEED_RefSpeedRpmToRad	Converts speed reference value from RPM to RAD
R_MOTOR_SPEED_VoltageStateSet	Sets the state that determines the output voltage.
R_MOTOR_SPEED_VoltageStateGet	Acquires the state that determines the output voltage. (Sensorless only)
R_MOTOR_SPEED_DrawInFlagSet	Sets the flag for pulling in the motor at startup. (Sensorless only)
R_MOTOR_SPEED_DrawInFlagGet	Acquires the flag for pulling in the motor at startup. (Sensorless only)

5.3.5 Configurations

Table 5-17 lists the configurations for the speed control module. Set up the functions to be used and the necessary parameters. Table 5-18 shows the initial values for each microcontroller.

Table 5-17 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	SPEED_CFG_SPEED_PI_DECIMATION	Number of PI decimations
	SPEED_CFG_SPEED_PI_KP	Proportional term of the PI gain for the speed control system
	SPEED_CFG_SPEED_PI_KI	Differential term of the PI gain for the speed control system
	SPEED_CFG_SPEED_PI_I_LIMIT_V	Differential term limit value of the PI gain for the speed control system
	SPEED_CFG_SPEED_CALC_BASE	Frequency setting of the speed calculation timer
	SPEED_CFG_SPEED_LPF_K	LPF gain for the speed control system
	SPEED_CFG_SPEED_CHANGE_LIMIT	Speed variation limit value
	MOTOR_CFG_MAX_OUTPUT_VOLTAGE	Maximum output voltage [V]
	MOTOR_CFG_MIN_OUTPUT_VOLTAGE	Minimum output voltage [V]
	MOTOR_CFG_START_REF_VOLTAGE	Command voltage at startup [V]
	MOTOR_CFG_MAX_BOOT_VOLTAGE	Maximum voltage in boot mode [V] (Sensorless only)
	MOTOR_CFG_SHIFT_ADJUST	Amount of phase adjustment (Sensorless only)

Table 5-18 List of initial values for configurations

Macro name	RX13T, RX23T, RX24T, RX24U
SPEED_CFG_SPEED_PI_DECIMATION	With Hall sensor: 0 Sensorless: 1
SPEED_CFG_SPEED_PI_KP	0.02f
SPEED_CFG_SPEED_PI_KI	With Hall sensor: 0.0005f Sensorless: 0.004f
SPEED_CFG_SPEED_PI_I_LIMIT_V	24.0f
SPEED_CFG_SPEED_CALC_BASE	MTR_TWOPI × 5000000
SPEED_CFG_SPEED_LPF_K	1.0f
SPEED_CFG_SPEED_CHANGE_LIMIT	With Hall sensor: 0.2f Sensorless: 0.2f * MTR_RPM2RAD
MOTOR_CFG_MAX_OUTPUT_VOLTAGE	20.0f
MOTOR_CFG_MIN_OUTPUT_VOLTAGE	With Hall sensor: 3.0f Sensorless: 5.0f
MOTOR_CFG_START_REF_VOLTAGE	5.8f
MOTOR_CFG_MAX_BOOT_VOLTAGE	8.0f (Sensorless only)
MOTOR_CFG_SHIFT_ADJUST	0 (Sensorless only)

5.3.6 Structure and variable information

Table 5-19 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 5-19 List of structures and variables (1)

Structure	Variable	Description
st_speed_control_t Structure for the speed control module	u2_run_mode	Run mode
	f4_v_ref	Voltage command value [V]
	u1_cnt_speed_pi	Count value for speed PI
	u1_flag_speed_ref	Flag for speed command value
	u1_flag_voltage_ref	Flag for voltage command value
	f4_ref_speed_rad	Speed command value [rad/s]
	f4_ref_speed_rad_ctrl	Speed command value for control [rad/s]
	f4_speed_rad	Speed [rad/s]
	f4_kp_speed	Proportional term gain for PI control
	f4_ki_speed	Differential term gain for PI control
	f4_limit_speed_change	Speed variation limit
	f4_ilim_v	Max voltage
	f4_start_refv	Command voltage at startup [V]
	f4_rpm_rad	Variable for unit conversion (rpm to rad/s)
	f4_boot_ref_v	Command voltage in boot mode [V] (Sensorless only)
	u2_v_up_time	Voltage rising duration [s] (Sensorless only)
	f4_v_up_step	Voltage rising range [V] (Sensorless only)
	u2_v_const_time	Constant voltage duration [s] (Sensorless only)
	u2_cnt_adj_v	Voltage rise time counter (Sensorless only)
	u1_flag_draw_in	Initial position setting count flag (Sensorless only)
	s2_ol_start_rad	Speed at open loop startup [rad/s] (Sensorless only)
	s2_ol_mode1_change_rad	Speed to transition to open loop mode 1 [rad/s] (Sensorless only)
	s2_ol_mode2_change_rad	Speed to transition to open loop mode 2 [rad/s] (Sensorless only)
	f4_ol_start_refv	Command voltage setting at open loop startup [V] (Sensorless only)
	f4_ol_mode1_rate_rad	The speed variation setting in open loop mode 1*1 [rad/s] (Sensorless only)
	f4_ol_mode2_rate_refv	The voltage variation setting in open loop mode 2*1 [V] (Sensorless only)
	f4_ol_mode2_rate_rad	The speed variation setting in open loop mode 2*1 [rad/s] (Sensorless only)
	f4_ol_mode3_rate_refv	The voltage variation setting in open loop mode 3*1 [V] (Sensorless only)
	st_pi_ctrl_t	Structure for PI control
	st_motor_parameter_t	Structure for motor parameters

Note: 1. For details, see 5.2.11, Startup sequence management.

Table 5-20 List of structures and variables (2)

Structure	Variable	Description
st_speed_config_t Structure for setting the parameters for controlling the speed control module	f4_ref_speed_rad	Command speed [rad/s]
	f4_kp_speed	Proportional term gain for PI control
	f4_ki_speed	Differential term gain for PI control
	f4_limit_speed_change	Speed variation limit
	f4_start_refv	Command voltage at startup [V]
	f4_rpm_rad	Variable for unit conversion (rpm to rad/s)
	f4_boot_ref_v	Command voltage in boot mode [V] (Sensorless only)
	u2_v_up_time	Voltage rising duration [s] (Sensorless only)
	u2_v_const_time	Voltage rising range [V] (Sensorless only)
	s2_ol_start_rad	Speed at open loop startup [rad/s] (Sensorless only)
	s2_ol_mode1_change_rad	Speed to transition to open loop mode 1 [rad/s] (Sensorless only)
	s2_ol_mode2_change_rad	Speed to transition to open loop mode 2 [rad/s] (Sensorless only)
	f4_ol_start_refv	Command voltage setting at open loop startup [V] (Sensorless only)
	f4_ol_mode1_rate_rad	The speed variation setting in open loop mode 1*[1] [rad/s] (Sensorless only)
	f4_ol_mode2_rate_refv	The voltage variation setting in open loop mode 2*[1] [V] (Sensorless only)
	f4_ol_mode2_rate_rad	The speed variation setting in open loop mode 2*[1] [rad/s] (Sensorless only)
	f4_ol_mode3_rate_refv	The voltage variation setting in open loop mode 3*[1] [V] (Sensorless only)
	f4_ol_mode3_max_refv	The maximum command voltage in open loop mode 3*[1] [V] (Sensorless only)
st_speed_input_t Structure for speed control module input	u2_run_mode	Run mode
	f4_speed_rad	Speed [rad/s]
st_speed_output_t Structure for speed control module output	f4_v_ref	Command voltage [V]

Note: 1. For details, see 5.2.11, Startup sequence management.

5.3.7 Macro definition

Table 5-21 lists the macros of the speed control module.

Table 5-21 List of macros

File name	Macro name	Defined value	Remarks
r_motor_speed_api.h	VOLTAGE_STATE_ZERO_CONST	0	The voltage command value is fixed to 0.
	VOLTAGE_STATE_UP	1	The voltage command value is increased.
	VOLTAGE_STATE_CONST	2	The voltage command value is fixed.
	VOLTAGE_STATE_OPENLOOP	3	The voltage command value is set to a fixed open loop value.
	VOLTAGE_STATE_PI_OUTPUT	4	The voltage command value is set to a PI control output value.
	SPEED_STATE_ZERO_CONST	0	The speed command value is fixed to 0.
	SPEED_STATE_OPENLOOP_1	1	The speed command value is set to open loop mode 1*[1].
	SPEED_STATE_OPENLOOP_2	2	The speed command value is set to open loop mode 2*[1].
	SPEED_STATE_OPENLOOP_3	3	The speed command value is set to open loop mode 3*[1].
	SPEED_STATE_SPEED_CHANGE	4	The speed command value is set to the user-specified value.

Note: 1. For details, see 5.2.11, Startup sequence management.

5.3.8 Adjustment and configuration of parameters

(a) Gain adjustment of the speed control system

In the target software for this application notes, speed control is performed by PI control. The following speed PI control is used to get the voltage command value. For details on how to set Kp and Ki, refer to a technical book on PI control.

$$v^{*} = (K_{P\omega} + \frac{K_{I\omega}}{s})(\omega^{*} - \omega)$$

v^{*} : *Voltage command value* ω^{*} : *Speed command value* ω: *Rotation speed*

$K_{P\omega}$: *Speed PI proportional gain* $K_{I\omega}$: *Speed PI integral gain* s: *Laplace operator*

(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 (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 5.3.5.

5.4 Sensor module (Hall sensor)

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

5.4.1 Functionality

Table 5-22 lists the functions of the sensor module.

Table 5-22 List of functions of the sensor module

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

5.4.2 Module configuration diagram

Figure 5-21 shows the sensor module configuration.

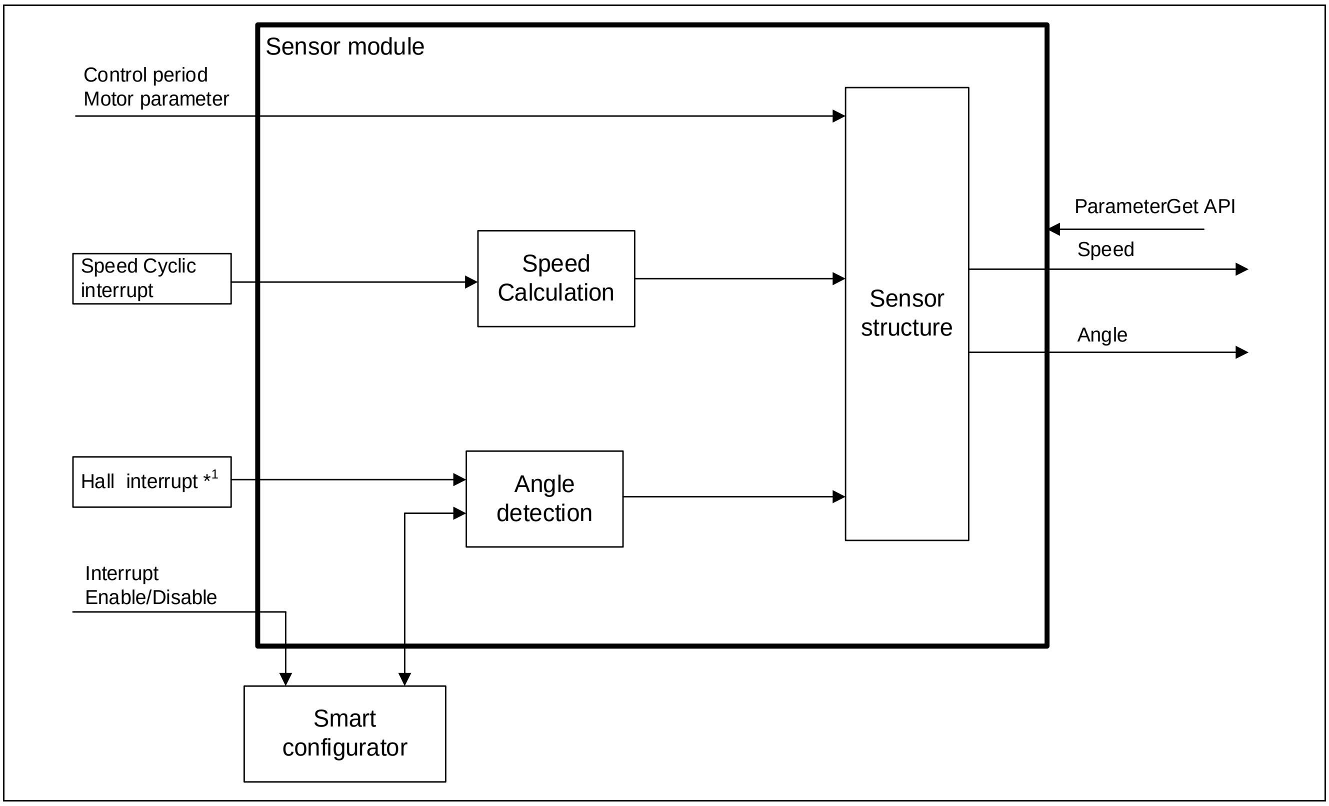


Figure 5-21 Sensor module configuration diagram

Note: 1. Carrier Interrupt when sensorless

5.4.3 Flowcharts

See 5.2.5(d) for a flowchart on Hall sensor interrupt processing.

5.4.4 API

Table 5-23 lists the API functions for the sensor module.

Table 5-23 List of API functions

API function	Description
R_MOTOR_SENSOR_Open	Generates an instance of the sensor module. Run this API function first when using this module.
R_MOTOR_SENSOR_Close	Places the sensor module in a reset state.
R_MOTOR_SENSOR_Reset	Initializes the module.
R_MOTOR_SENSOR_ParameterSet	Sets the parameters used in the sensor module.
R_MOTOR_SENSOR_ParameterGet	Acquires the parameters of the sensor module.
R_MOTOR_SENSOR_ParameterUpdate	Updates the control parameters of the sensor module.
R_MOTOR_SENSOR_PatternSet	Sets the conduction pattern. (Only when a Hall sensor is used)
R_MOTOR_SENSOR_DirectionSet	Sets the direction of motor rotation.
R_MOTOR_SENSOR_DirectionGet	Acquires the direction of motor rotation.
R_MOTOR_SENSOR_TimerCountGet	Motor drive confirmation timer counter value acquisition.
R_MOTOR_SENSOR_TimerCountUp	Motor drive confirmation timer count up.
R_MOTOR_SENSOR_TimerCountClear	Motor drive confirmation timer clear the counter.
R_MOTOR_SENSOR_StartOpenLoop	Starts open loop drive. (Sensorless only)
R_MOTOR_SENSOR_AngleShiftSet	Sets the phase shift information for switching the conduction pattern. (Sensorless only)
R_MOTOR_SENSOR_CheckPattern	Checks for errors in the conduction pattern.
R_MOTOR_SENSOR_ShiftAngle	Sets the conduction pattern and switches the conduction pattern. (Sensorless only)
R_MOTOR_SENSOR_OutputPatternSet	Sets the conduction pattern and switches the conduction pattern. (Sensorless only)

5.4.5 Configurations

Table 5-24 lists the configurations for the sensor module. Set up the functions to be used and the necessary parameters. Table 5-25 shows the initial values for each microcontroller.

Table 5-24 List of configurations

File name	Macro name	Description
r_motor_module_cfg.h	SENSOR_CFG_HALL_TIMEOUT_CNT	Hall timeout count time
	SENSOR_CFG_HALL_STOP_WAIT_CNT	Count value for stop detection
	SENSOR_CFG_HALL_WAIT_SPEED_CALC	Waiting time for speed calculation until a Hall interrupt occurs

Table 5-25 List of initial values for configurations

Macro name	RX13T, RX23T, RX24T, RX24U
SENSOR_CFG_HALL_TIMEOUT_CNT	With Hall sensor: 4000 Sensorless: 2000
SENSOR_CFG_HALL_STOP_WAIT_CNT	4000
SENSOR_CFG_HALL_WAIT_SPEED_CALC	12

5.4.6 Structure and variable information

Table 5-26 lists the structures and variables for the sensor module.

Table 5-26 List of structures and variables

Structure	Variable	Description
st_sensor_t Structure for the sensor module	u1_mode_system	System mode
	u2_run_mode	Run mode
	u2_error_status	Error status
	u1_v_pattern	Conduction pattern
	u2_pre_hall_timer_cnt	Previous value of the Hall timer
	u2_hall_timer_cnt	Count value of the Hall timer
	s4_timer_cnt_ave	Average of speed measurement timer counts
	u2_timer_cnt_buf[6]	Buffer for speed measurement
	u2_timer_cnt_num	Count number for speed measurement
	u1_hall_signal	Hall signal
	u1_hall_wait_cnt	Count value for the Hall wait time
	u1_direction	Rotation direction
	u2_pwm_duty	PWM duty value
	f4_speed_rad	Speed [rad/s]
	f4_speed_lpf_k	LPF gain for speed
	u2_cnt_timeout	Timer counter for sensor timeout detection
	u1_v_pattern_num	Conduction pattern number (Sensorless only)
	u1_bemf_signal	Signal value of counter-electromotive force (Sensorless only)
	u1_pre_bemf_signal	Previous counter-electromotive force signal value. (Sensorless only)
	u1_flag_pattern_change	Flag for switching the conduction pattern (Sensorless only)
	u1_v_pattern_ol[2][7]	Conduction pattern during open loop control (Sensorless only)
	u1_ol_signal	Counter-electromotive force during open loop control (Sensorless only)
	u2_ol_pattern_set	Conduction pattern setting during open loop control (Sensorless only)
	u2_cnt_ol_pattern_set	Conduction pattern setting count value during open loop control (Sensorless only)
	u2_bemf_timer_cnt	Timer count value of counter-electromotive force (Sensorless only)
	u2_pre_bemf_timer_cnt	Previous timer value of counter-electromotive force (Sensorless only)
	u2_cnt_carrier	Count value for carrier wave interrupts (Sensorless only)
	u2_pre_cnt_carrier	Previous value for carrier wave interrupts (Sensorless only)
	u2_angle_shift_cnt	Count value for phase shift (Sensorless only)
	s2_angle_shift_adjust	Angle phase adjustment (Sensorless only)
st_sensor_output_t Structure for sensor module output	f4_speed_rad	Speed [rad/s]
	u2_error_status	Error status

Structure	Variable	Description
st_sensor_input_t Structure for sensor module input	u1_mode_system	System mode
	u2_run_mode	Run mode
	u2_pwm_duty	PWM duty value
st_sensor_cfg_t Structure for setting the parameters of the sensor module	f4_speed_lpf_k	LPF gain for speed
	u1_hall_wait_cnt	Count value for the Hall wait time

5.4.7 Macro definition

Table 5-27 lists the macros for the sensor module.

Table 5-27 List of macros

File name	Macro name	Defined value	Remarks
r_motor_sensor.h	SENSOR_PATTERN_CW_V_U	(2)	Conduction pattern during CW rotation (from V-phase to U-phase)
	SENSOR_PATTERN_CW_W_U	(3)	Conduction pattern during CW rotation (from W-phase to U-phase)
	SENSOR_PATTERN_CW_W_V	(1)	Conduction pattern during CW rotation (from W-phase to V-phase)
	SENSOR_PATTERN_CW_U_V	(5)	Conduction pattern during CW rotation (from U-phase to V-phase)
	SENSOR_PATTERN_CW_U_W	(4)	Conduction pattern during CW rotation (from U-phase to W-phase)
	SENSOR_PATTERN_CW_V_W	(6)	Conduction pattern during CW rotation (from V-phase to W-phase)
	SENSOR_PATTERN_CCW_V_U	(5) (3)	Conduction pattern during CCW rotation (from V-phase to U-phase) Hall sensor :(5) Sensorless: (3)
	SENSOR_PATTERN_CCW_V_W	(1) (2)	Conduction pattern during CCW rotation (from V-phase to W-phase) Hall sensor :(1) Sensorless: (2)
	SENSOR_PATTERN_CCW_U_W	(3) (6)	Conduction pattern during CCW rotation (from U-phase to W-phase) Hall sensor :(3) Sensorless: (6)
	SENSOR_PATTERN_CCW_U_V	(2) (4)	Conduction pattern during CCW rotation (from U-phase to V-phase) Hall sensor :(2) Sensorless: (4)
	SENSOR_PATTERN_CCW_W_V	(6) (5)	Conduction pattern during CCW rotation (from W-phase to V-phase) Hall sensor :(6) Sensorless: (5)
	SENSOR_PATTERN_CCW_W_U	(4) (1)	Conduction pattern during CCW rotation (from W-phase to U-phase) Hall sensor :(4) Sensorless: (1)
	SENSOR_PATTERN_ERROR	(0)	Macro for conduction pattern error
	SENSOR_UP_PWM_VN_ON	(1)	The upper arm of U-phase is PWM output, and the lower arm of V-phase is in conduction.
	SENSOR_UP_PWM_WN_ON	(2)	The upper arm of U-phase is PWM output, and the lower arm of W-phase is in conduction.
	SENSOR_VP_PWM_UN_ON	(3)	The upper arm of V-phase is PWM output, and the lower arm of U-phase is in conduction.
	SENSOR_VP_PWM_WN_ON	(4)	The upper arm of V-phase is PWM output, and the lower arm of W-phase is in conduction.
	SENSOR_WP_PWM_UN_ON	(5)	The upper arm of W-phase is PWM output, and the lower arm of U-phase is in conduction.
	SENSOR_WP_PWM_VN_ON	(6)	The upper arm of W-phase is PWM output, and the lower arm of V-phase is in conduction.
	SENSOR_UP_ON_VN_PWM	(7)	The upper arm of U-phase is in conduction, and the lower arm of V-phase is PWM output.

File name	Macro name	Defined value	Remarks
r_motor_sensor.h	SENSOR_UP_ON_WN_PWM	(8)	The upper arm of U-phase is in conduction, and the lower arm of W-phase is PWM output.
	SENSOR_VP_ON_UN_PWM	(9)	The upper arm of V-phase is in conduction, and the lower arm of U-phase is PWM output.
	SENSOR_VP_ON_WN_PWM	(10)	The upper arm of V-phase is in conduction, and the lower arm of W-phase is PWM output.
	SENSOR_WP_ON_UN_PWM	(11)	The upper arm of W-phase is in conduction, and the lower arm of U-phase is PWM output.
	SENSOR_WP_ON_VN_PWM	(12)	The upper arm of W-phase is in conduction, and the lower arm of V-phase is PWM output.
	SENSOR_U_PWM_VN_ON	(13)	U-phase is PWM output, and the lower arm of V-phase is in conduction.
	SENSOR_U_PWM_WN_ON	(14)	U-phase is PWM output, and the lower arm of W-phase is in conduction.
	SENSOR_V_PWM_UN_ON	(15)	V-phase is PWM output, and the lower arm of U-phase is in conduction.
	SENSOR_V_PWM_WN_ON	(16)	V-phase is PWM output, and the lower arm of W-phase is in conduction.
	SENSOR_W_PWM_UN_ON	(17)	W-phase is PWM output, and the lower arm of U-phase is in conduction.
	SENSOR_W_PWM_VN_ON	(18)	W-phase is PWM output, and the lower arm of V-phase is in conduction.
	SENSOR_UP_ON_V_PWM	(19)	The upper arm of U-phase is in conduction, and V-phase is PWM output.
	SENSOR_UP_ON_W_PWM	(20)	The upper arm of U-phase is in conduction, and W-phase is PWM output.
	SENSOR_VP_ON_U_PWM	(21)	The upper arm of V-phase is in conduction, and U-phase is PWM output.
	SENSOR_VP_ON_W_PWM	(22)	The upper arm of V-phase is in conduction, and W-phase is PWM output.
	SENSOR_WP_ON_U_PWM	(23)	The upper arm of W-phase is in conduction, and U-phase is PWM output.
	SENSOR_WP_ON_V_PWM	(24)	The upper arm of W-phase is in conduction, and V-phase is PWM output.

5.4.8 Adjustment and configuration of parameters

The initial values of sensor module parameters can be specified with the configuration information (r_motor_module_cfg.h). The specified configurations are applied when the system starts. For details about the items to be set, see 5.4.5.

5.4.9 How to calculate the speed

(a) Hall sensor

For a configuration with a Hall sensor, the motor rotation speed is calculated by letting the channel 1 timer of the compare match timer run free, capture the timer value through the external interrupt routine triggered by Hall sensor signals, and then obtaining the difference between the current timer value and the timer value 2π [rad] earlier. A low pass filter (LPF) is applied to the speed calculation result.

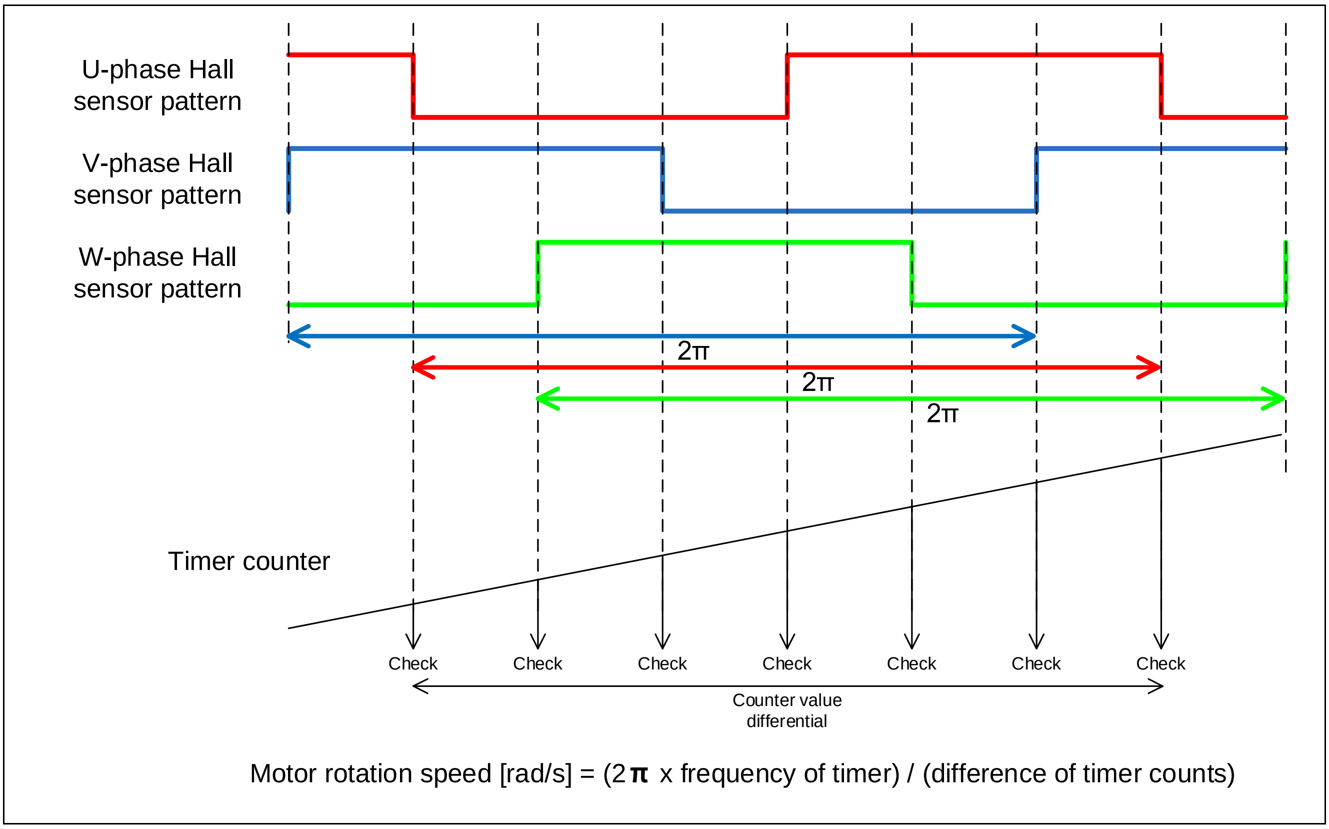


Figure 5-22 How to calculate the motor rotation speed

(b) Sensorless

For a sensorless configuration, the motor rotation speed is calculated by letting the channel 1 timer of the compare match timer run free, capture the timer value at zero crossings, and then obtaining the difference between the current timer value and the timer value 2π [rad] earlier.

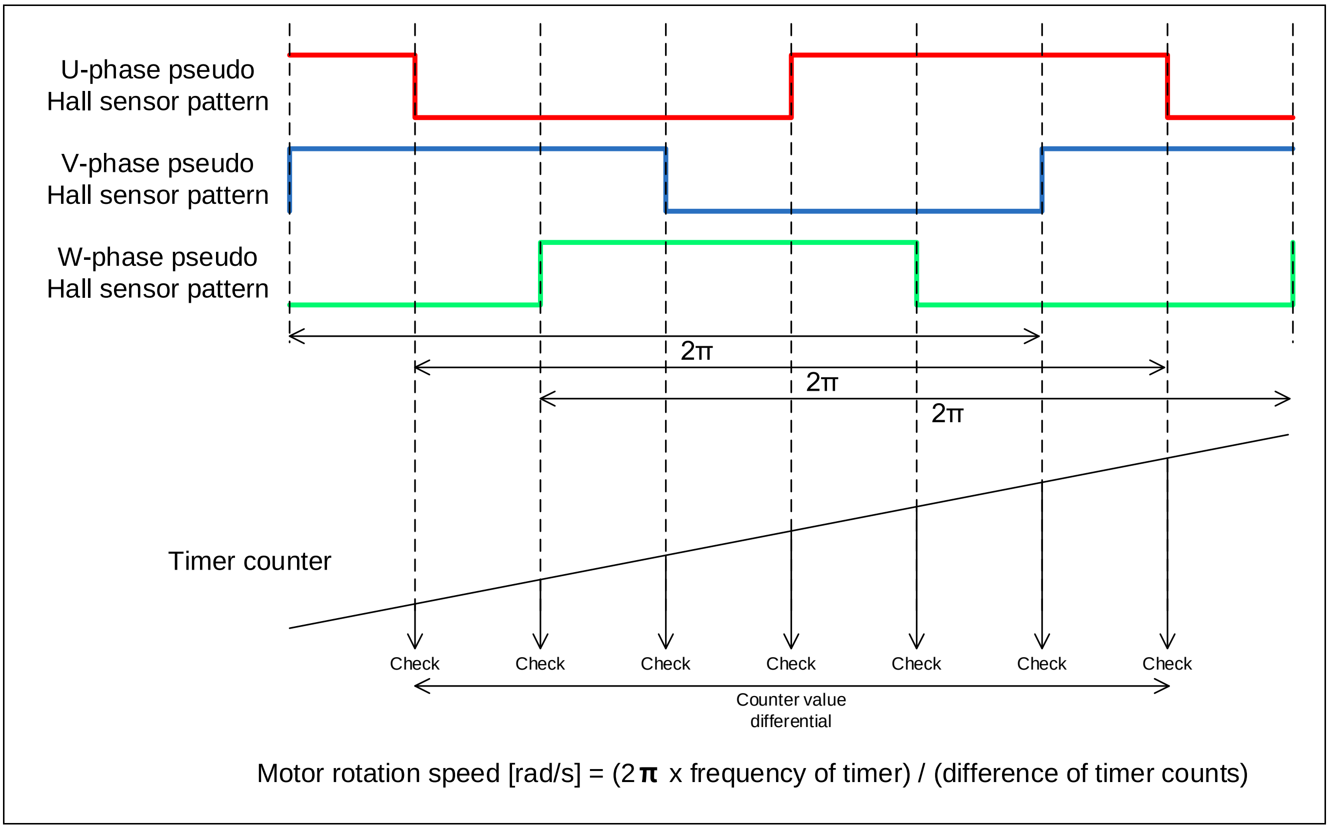


Figure 5-23 How to calculate the motor rotation speed

5.5 Driver module

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

5.5.1 Functionality

Table 5-28 lists the functions of the driver module.

Table 5-28 List of functions of the driver module

Function	Description
Acquisition of the A/D conversion value	Acquires AD values such as the phase current , phase voltage *1 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.
Speed measurement timer start/stop	Sets whether to start or stop the speed measurement timer via an Smart Configurator function.
Hall sensor interrupts enable/disable	Sets whether to enable or disable Hall sensor interrupts via an Smart Configurator function.

Note: 1. Phase voltage is acquired only when there is no sensor

5.5.2 Module configuration diagram

Figure 5-24 Driver module configuration diagram shows the module configuration of the driver module.

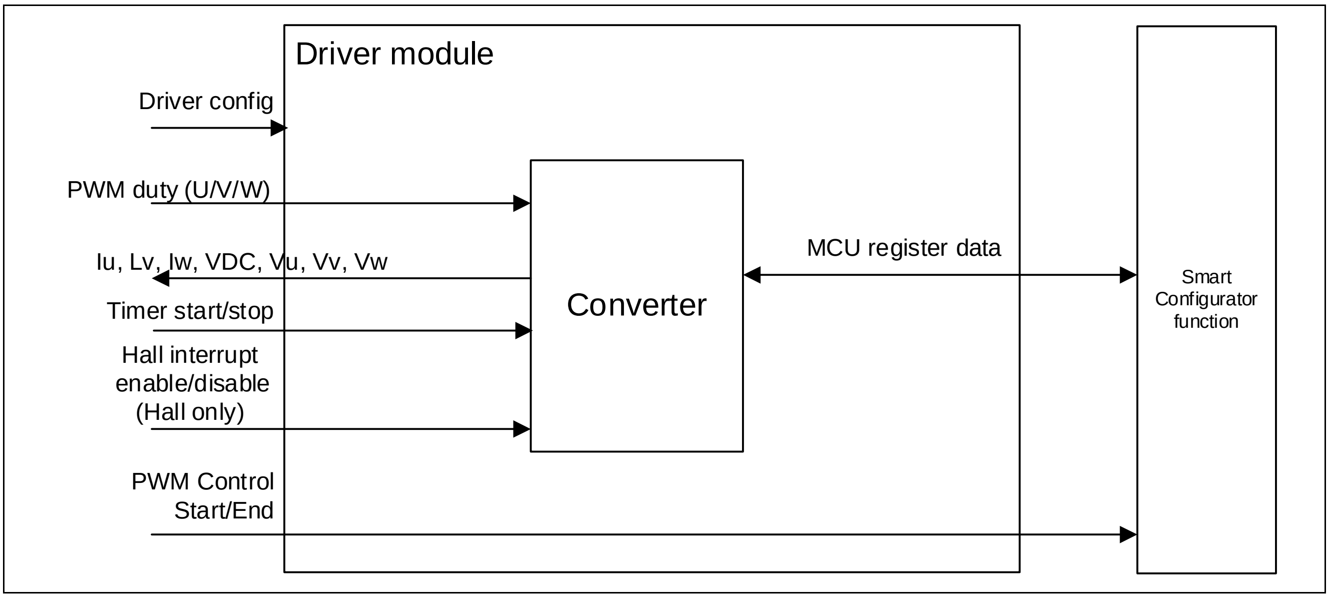


Figure 5-24 Driver module configuration diagram

5.5.3 API

Table 5-29 lists and describes the API functions for the driver module.

Table 5-29 List of API functions

API	Description
R_MOTOR_DRIVER_Open	Generates an instance of the driver module.
R_MOTOR_DRIVER_Close	Places the module in a reset state.
R_MOTOR_DRIVER_ParameterUpdate	Inputs the variable information that is to be used inside the module.
R_MOTOR_DRIVER_BldcAnalogGet	Acquires the A/D conversion results.
R_MOTOR_DRIVER_PWMControlStop	Stops PWM control.
R_MOTOR_DRIVER_PWMControlStart	Starts PWM control.
R_MOTOR_DRIVER_OutputPatternChange	Switches the 120-degree conducting control conduction pattern.

5.5.4 Configurations

Table 5-30 lists the configurations for the driver 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	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_CHANGE_PATTERN	Sets the function that changes the conduction pattern.
	DRIVER_CFG_FUNC_FREERUN_TIMER_START	Set the function that starts the free-run timer for speed calculation.
	DRIVER_CFG_FUNC_FREERUN_TIMER_STOP	Set the function that stops the free-run timer for speed calculation.
	DRIVER_CFG_FUNC_FREERUN_TIMERCNT_SET	Set the function that sets the count value of the free-run timer for speed calculation.
	DRIVER_CFG_FUNC_FREERUN_TIMERCNT_GET	Set the function that acquires the count value of the free-run timer for speed calculation.
	DRIVER_CFG_HALL_FUNC_INT_U_START	Set the function that enables U-phase interrupts of the Hall sensor.
	DRIVER_CFG_HALL_FUNC_INT_V_START	Set the function that enables V-phase interrupts of the Hall sensor.
	DRIVER_CFG_HALL_FUNC_INT_W_START	Set the function that enables W-phase interrupts of the Hall sensor.
	DRIVER_CFG_HALL_FUNC_INT_U_STOP	Set the function that disables U-phase interrupts of the Hall sensor.
	DRIVER_CFG_HALL_FUNC_INT_V_STOP	Set the function that disables V-phase interrupts of the Hall sensor.
	DRIVER_CFG_HALL_FUNC_INT_W_STOP	Set the function that disables W-phase interrupts of the Hall sensor.

Table 5-31 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_CHANGE_PATTERN	R_Config_xxx_Chg_Pattern (Smart Configurator function) *1 *2
DRIVER_CFG_FUNC_FREERUN_TIMER_START	R_Config_CMTx_Start (Smart Configurator function) *1
DRIVER_CFG_FUNC_FREERUN_TIMER_STOP	R_Config_CMTx_Stop (Smart Configurator function) *1
DRIVER_CFG_FUNC_FREERUN_TIMERCNT_SET	R_Config_CMTx_Set_Cmcnt (Smart Configurator function) *1
DRIVER_CFG_FUNC_FREERUN_TIMERCNT_GET	R_Config_CMTx_Get_Cmcnt (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_U_START	R_Config_ICU_IRQx_Start (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_V_START	R_Config_ICU_IRQx_Start (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_W_START	R_Config_ICU_IRQx_Start (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_U_STOP	R_Config_ICU_IRQx_Stop (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_V_STOP	R_Config_ICU_IRQx_Stop (Smart Configurator function) *1
DRIVER_CFG_HALL_FUNC_INT_W_STOP	R_Config_ICU_IRQx_Stop (Smart Configurator function) *1

Notes: 1. For details about the functions shown in the "Set value" column, see 5.6 Smart Configurator settings.
2. When the Smart Configurator Motor component is used, “xxx” is set to “MOTOR”. When the Motor component is not used, it is the module name used for PWM.

5.5.5 Structure and variable information

Table 5-32 lists the structures that are used for the driver module. In the driver module, the structure for the driver module (g_st_driver) is defined by securing an instance of the module from the API.

Table 5-32 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.)
	*ChangeOutputPattern	Pointer to the Smart Configurator function (This variable sets the function that switches the Hall pattern.)
	*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.)
	*FreerunTimerStart	Pointer to the Smart Configurator function (This variable sets the function that starts the speed measurement timer.)
	*FreerunTimerStop	Pointer to the Smart Configurator function (This variable sets the function that stops the speed measurement timer.)
	*FreerunTimerSet	Pointer to the Smart Configurator function (This variable sets the function that sets the value of the speed measurement timer.)
	*FreerunTimerGet	Pointer to the Smart Configurator function (This variable sets the function that acquires the value of the speed measurement timer.)
	*HallUEnable	Pointer to the Smart Configurator function (This variable sets the function that enables U-phase Hall interrupts.)
	*HallVEnable	Pointer to the Smart Configurator function (This variable sets the function that enables V-phase Hall interrupts.)
	*HallWEnable	Pointer to the Smart Configurator function (This variable sets the function that enables W-phase Hall interrupts.)
	*HallUDisable	Pointer to the Smart Configurator function (This variable sets the function that disables U-phase Hall interrupts.)
	*HallVDisable	Pointer to the Smart Configurator function (This variable sets the function that disables U-phase Hall interrupts.)
	*HallWDisable	Pointer to the Smart Configurator function (This variable sets the function that disables U-phase Hall interrupts.)
st_motor_driver_cfg_t Structure for setting the parameters for controlling the drive module	Same as st_motor_driver_t	Structure used to set the driver module

5.5.6 Macro definition

There are no macros defined by the driver.

5.5.7 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 5-33 shows the settings.

Table 5-33 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.
*ChangeOutputPattern	DRIVER_CFG_FUNC_CHANGE_PATTERN	
*PWMOutputStop	DRIVER_CFG_FUNC_PWM_OUTPUT_START	
*PWMOutputStart	DRIVER_CFG_FUNC_PWM_OUTPUT_STOP	
*FreerunTimerStart	DRIVER_CFG_FUNC_FREERUN_TIMER_START	
*FreerunTimerStop	DRIVER_CFG_FUNC_FREERUN_TIMER_STOP	
*FreerunTimerSet	DRIVER_CFG_FUNC_FREERUN_TIMERCNT_SET	
*FreerunTimerGet	DRIVER_CFG_FUNC_FREERUN_TIMERCNT_GET	
*HallUEnable	DRIVER_CFG_HALL_FUNC_INT_U_START	
*HallVEnable	DRIVER_CFG_HALL_FUNC_INT_V_START	
*HallWEnable	DRIVER_CFG_HALL_FUNC_INT_W_START	
*HallUDisable	DRIVER_CFG_HALL_FUNC_INT_U_STOP	
*HallVDisable	DRIVER_CFG_HALL_FUNC_INT_V_STOP	
*HallWDisable	DRIVER_CFG_HALL_FUNC_INT_W_STOP	

5.6 Smart Configurator settings

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

5.6.1 Clock settings

Table 5-34 shows the clock settings.

Table 5-34 MCU clock settings

Clock	Frequency		
	RX13T	RX23T	RX24T, RX24U
Main clock	32MHz *1	10MHz	10MHz
System clock (ICLK)	32MHz	40MHz	80MHz
Peripheral module clock (PCLKA)	-	40MHz	80MHz
Peripheral module clock (PCLKB)	32MHz	40MHz	40MHz
Peripheral module clock (PCLKC)	-	-	-
Peripheral module clock (PCLKD)	32MHz	40MHz	40MHz
External bus clock (BCLK)	-	-	-
Flash IF clock (FCLK)	32MHz	20MHz	20MHz
IWDTCLK	15kHz		

Note: 1. Set 32MHz using HOCO.

5.6.2 Component settings

Table 5-36 lists the components used and the functions allocated to the components.

Table 5-35 Smart Configurator components and their functions

Function	Component		
	RX13T	RX23T	RX24T, RX24U
Hall interrupt processing	Config_ICU		
3-phase PWM output, A/D conversion of current detection	Config_MOTOR		
A/D conversion processing (inverter bus voltage detection, command voltage detection for the board UI)	Config_S12AD0		Config_S12AD2
Setting of the port to be used	Config_PORT		
Position and speed control interrupt timer	Config_CMT0		
Free-run timer for speed measurement	Config_CMT1		
Independent watchdog timer	Config_IWDT		
Overcurrent detection	Config_POE		

5.6.3 Interrupts

Table 5-36 shows the information about the interrupts used for the MCUs that use the Motor component.

Table 5-36 List of interrupts

Component	Interrupt function	Description
Config_ICU	r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1 r_Config_ICU_irqxx_interrupt*1	Interrupts of the Hall sensor Interrupt level: 13 Multiple interrupt: Disabled
Config_MOTOR	r_Config_MOTOR_ad_interrupt	A/D conversion end interrupt Interrupt level: 14 Multiple interrupt: Enabled
Config_S12AD0/1	None	None
Config_PORT	None	None
Config_CMT0	r_Config_CMT0_cmi0_interrupt	Position and speed control interrupt Interrupt level: 13 Multiple interrupt: Enabled
Config_CMT1	None	None
Config_IWDT	None	None
Config_POE	r_Config_POE_oei1_interrupt	Hardware overcurrent interrupt Interrupt level: 15 Multiple interrupt: Disabled

Note: 1. The xx portion is the setting assigned to each MCU. For details, see Table 5-35.

5.6.4 Details of user codes

Table 5-37 lists the functions that are created in the user code area.

Table 5-37 List of functions in the user area

Component	Function	Description
Config_PORT	R_Config_PORT_GetSW1	Acquires the status of SW1.
	R_Config_PORT_GetSW2	Acquires the status of SW2.
	R_Config_PORT_Led1_on	Turns on LED1.
	R_Config_PORT_Led2_on	Turns on LED2.
	R_Config_PORT_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.
Config_MOTOR	R_Config_MOTOR_Change_Pattern	Switches the conduction pattern.

5.6.5 Pin settings

Table 5-38 shows the pin interface information.

Table 5-38 Pin interface

Function	RX13T	RX23T	RX24T	RX24U
Inverter Measurement of the bus voltage	P46/AN006	P43/AN003	P64/AN204	P64/AN204
Pin for inputting the position/speed command value (analog value)	P47/AN007	P10/AN017	P53/AN209	P53/AN209
START/STOP Toggle switch	PB5	P91	P80	P80
ERROR RESET Toggle switch	PB4	P92	P81	P81
LED1 control	PD6	P00	PA2	PA2
LED2 control	PD4	P01	PA1	PA1
LED3 control	PB3	P31	PD7	PD7
Measurement of the U-phase current	P40/AN000	P40/AN000	P44/AN100	P44/AN100
Measurement of the V-phase current	P41/AN001	P41/AN001	P45/AN101	P45/AN101
Measurement of the W-phase current	P42/AN002	P42/AN002	P46/AN102	P46/AN102
PWM output (U_p)	P71/MTIOC3B	P71/MTIOC3B	P71/MTIOC3B	P71/MTIOC3B
PWM output (V_p)	P72/MTIOC4A	P72/MTIOC4A	P72/MTIOC4A	P72/MTIOC4A
PWM output (W_p)	P73/MTIOC4B	P73/MTIOC4B	P73/MTIOC4B	P73/MTIOC4B
PWM output (U_n)	P74/MTIOC3D	P74/MTIOC3D	P74/MTIOC3D	P74/MTIOC3D
PWM output (V_n)	P75/MTIOC4C	P75/MTIOC4C	P75/MTIOC4C	P75/MTIOC4C
PWM output (W_n)	P76/MTIOC4D	P76/MTIOC4D	P76/MTIOC4D	P76/MTIOC4D
Hall U-phase input	P93/IRQ0	P93/IRQ0	P10/IRQ0	P10/IRQ0
Hall V-phase input	P94/IRQ1	P94/IRQ1	P11/IRQ1	P11/IRQ1
Hall W-phase input	PA2/IRQ4	PA2/IRQ4	P96/IRQ4	P01/IRQ4
Measurement of the U-phase voltage	P43/AN003	P44/AN004	P61/AN201	P61/AN201
Measurement of the V-phase voltage	P44/AN004	P45/AN005	P62/AN202	P62/AN202
Measurement of the W-phase voltage	P45/AN005	P46/AN006	P63/AN203	P63/AN203
PWM emergency stop input when an overcurrent is detected	PE2/POE10#	P70/POE0#	P70/POE0#	P70/POE0#

6. 120-degree conducting control algorithm

6.1 120-degree conducting control

If the conduction patterns of each phase are changed at every 60 degrees as shown in Figure 6-1, a torque is generated between coil flux and permanent magnet of a rotor and the rotor rotates synchronously with the flux. As a conduction session of each switching element is 120 degrees, this control method is referred to as 120-degree conducting control.

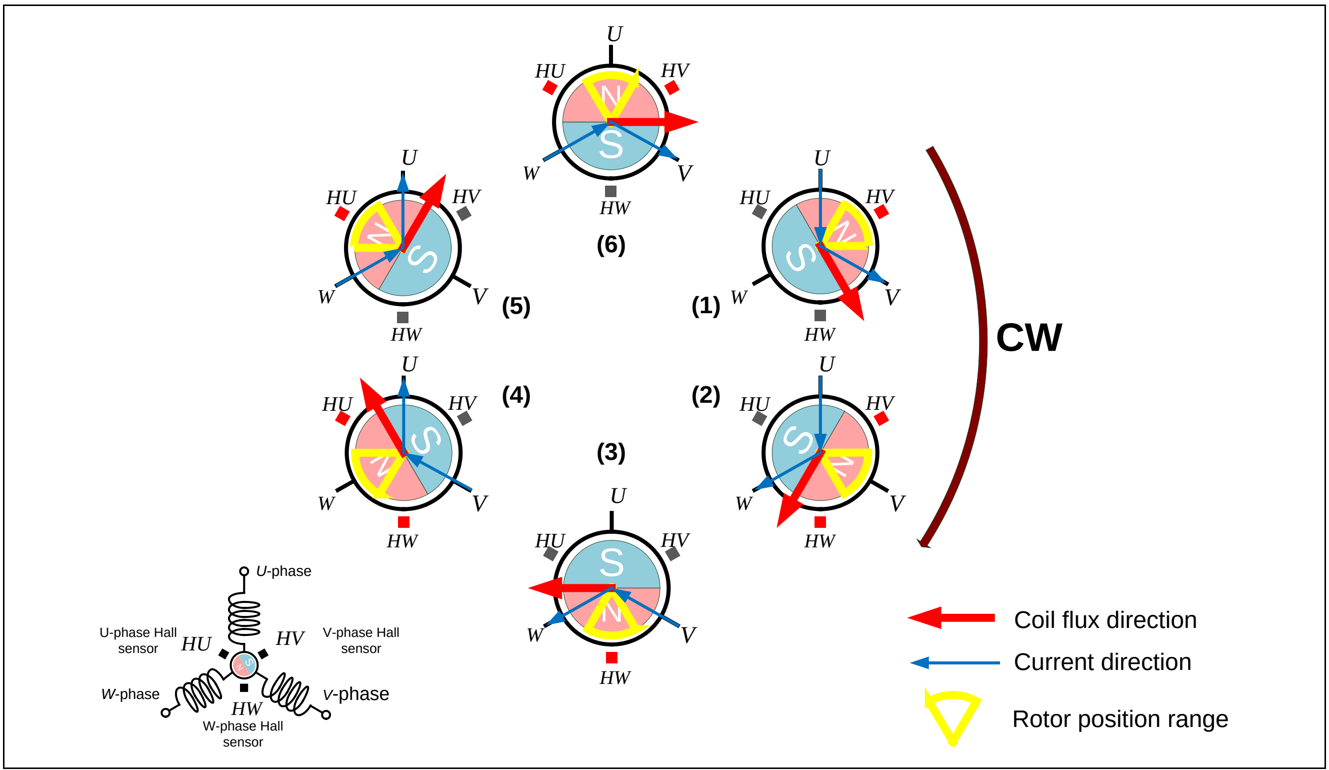


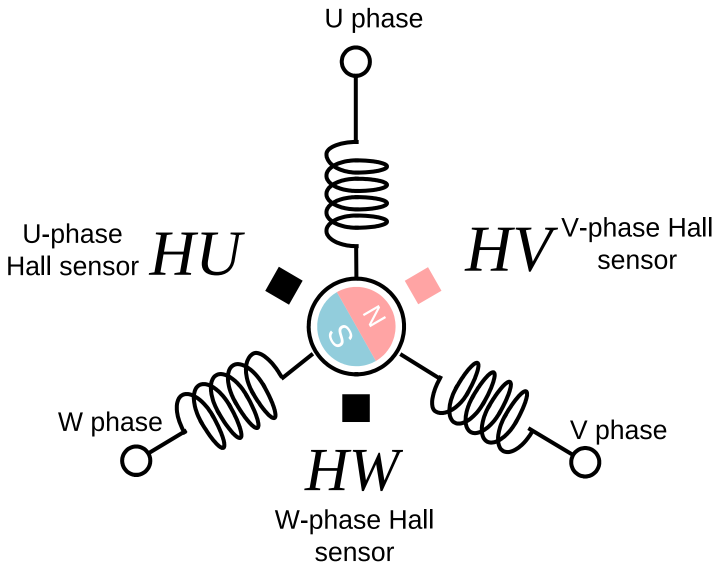
Figure 6-1 Six conduction patterns and rotor position ranges (example)

6.2 Position detection/speed calculation at 120-degree conducting control

6.2.1 120-degree conducting control using Hall sensors

(a) Position detection

The Hall sensors are used to detect the position of the permanent magnet, and the signals from the Hall sensors are inputted to the microcontroller as position information.



Hall sensor	HU	HV	HW
Signal	L	H	L

Figure 6-2 Example of Hall sensors (HU, HV, HW) position and signals

As shown in Figure 6-2, the Hall sensors are allocated every 120 degrees and the respective Hall sensor signals are switched depending on change in magnetic poles of the permanent magnet. Combining these signals of three Hall sensors enables to obtain position information every 60 degrees (six patterns for one cycle). At the switching timing of Hall sensor signals, the conduction patterns of each phase are changed as shown in Figure 6-3.

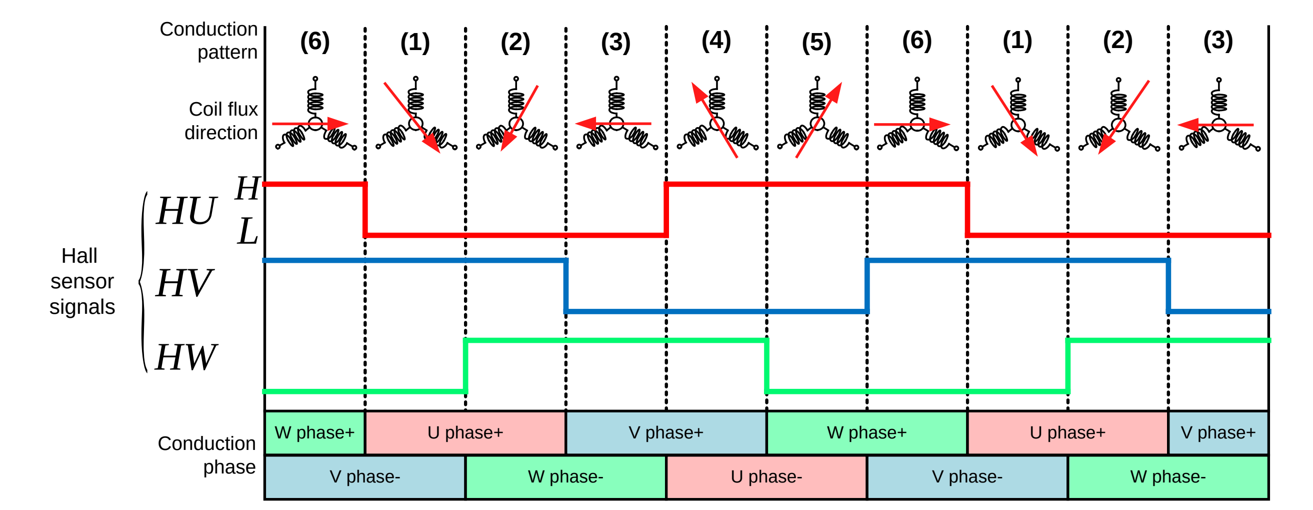


Figure 6-3 Relation between Hall sensor signals and conduction patterns (Rotation direction: CW)

(b) Speed calculation

The motor rotational speed can be calculated from a difference between the current timer value and the timer value 2π [rad] before. The timer values are obtained through the external interrupt routine by Hall sensor signals while having the peripheral function timer of the microcontroller performed free-running. With this method, the rotational speed can be calculated even when the Hall sensors are placed unequally.

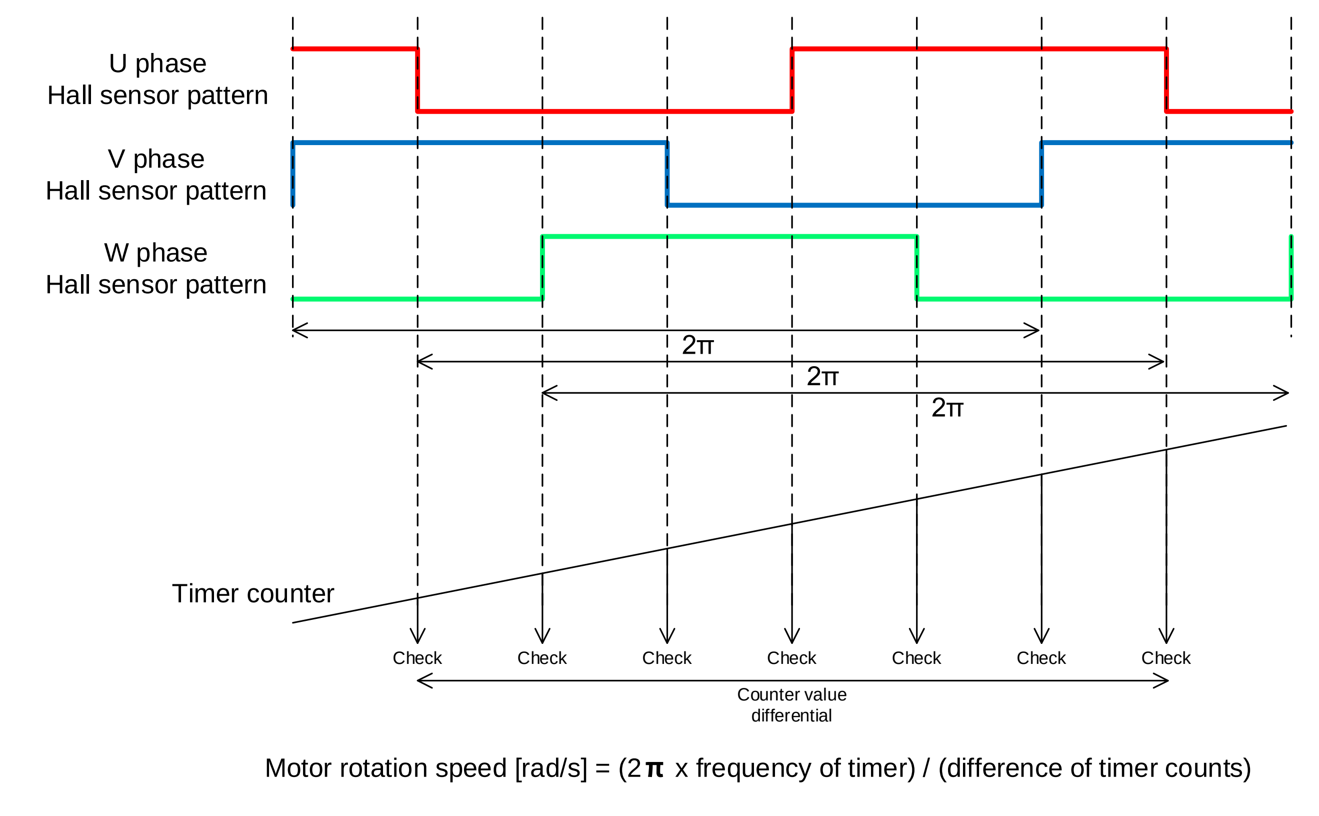


Figure 6-4 Method of calculation for the rotational speed

6.2.2 Sensorless 120-degree conducting control

(a) Position estimation

The sensorless control does not have a sensor for obtaining the permanent magnetic position, and hence an alternative to the sensor is required. The sensorless control of permanent magnetic synchronous motors generally estimates the position by detecting induced voltage.

The induced voltage is generated in proportion to a change rate of magnetic flux passing through a coil, to prevent the change.

For example, consider the case where a magnet gets close to the coil, as shown in Figure 6-5. In this case, since the magnetic flux increases within the coil, the coil generates the electromotive force that runs current in the direction of the figure to prevent the increase of magnetic flux. (The flux of opposite direction of the magnetic flux is generated by the right-handed screw rule.)

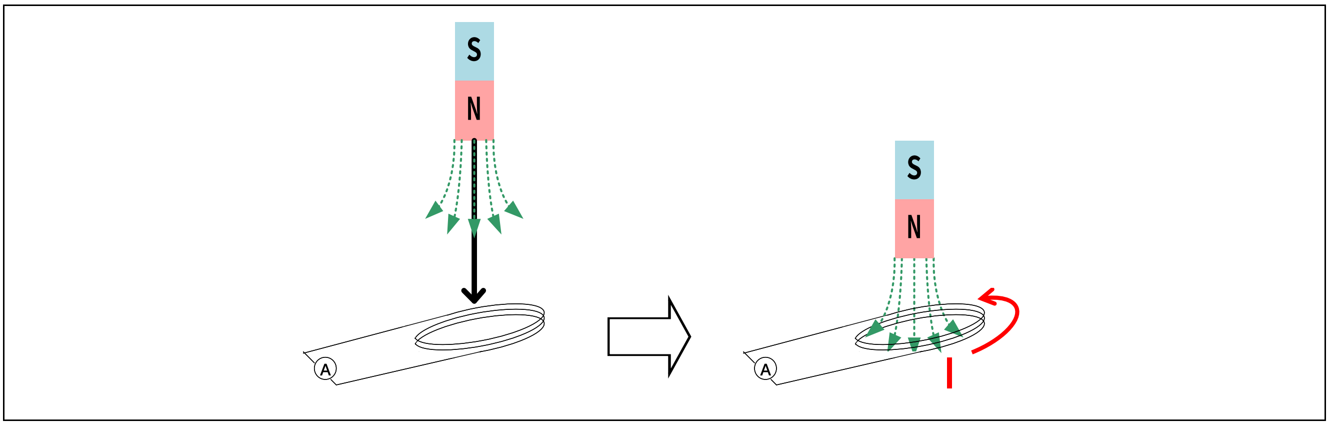


Figure 6-5 Induced voltage generated by coil and magnet

This induced voltage Em is expressed by the magnetic flux m as the below formula.

$$E_m = \frac{d}{dt}\varphi_m \cdots (1)$$

This phenomenon also occurs in a rotating permanent magnetic synchronous motor. When the permanent magnet is rotating, the induced voltage is generated by constant change of interlinkage magnetic flux of each phase.

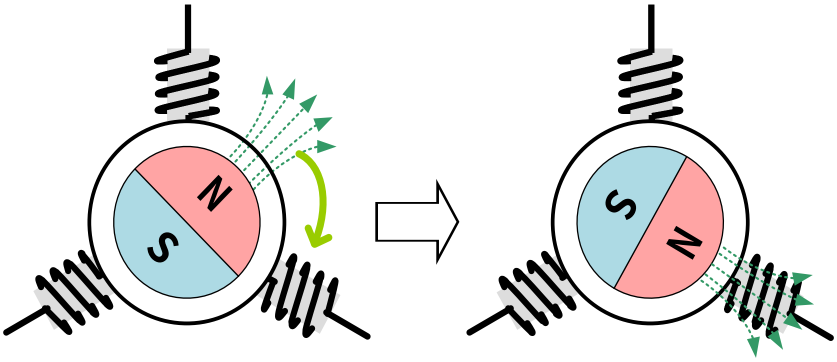


Figure 6-6 Induced voltage in the rotating permanent magnetic synchronous motor

Figure 6-7 shows the change of interlinkage magnetic flux in the U phase. Size of the interlinkage magnetic flux is shown on the vertical axis and the phase of the permanent magnet is shown on the horizontal axis. Also, a position where the N pole of the permanent magnet points the coil of the U phase is defined as θ = 0.

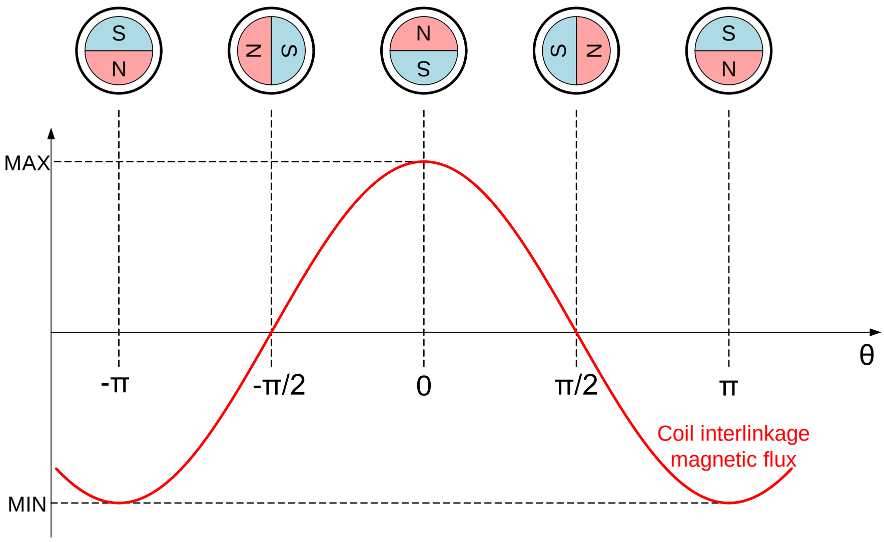


Figure 6-7 Change of the interlinkage magnetic flux

The interlinkage magnetic flux of the U phase changes in a cosine wave format.

If considered in same way about the V phase and W phase, they deviate respectively by 2π/3 and -2π/3 phase from the U phase. The interlinkage magnetic fluxes of the three phases are expressed by the following formula.

$$\varphi_u = \varphi_m \cos\theta$$

$$\varphi_v = \varphi_m \cos(\theta - \frac{2}{3}\pi)$$

$$\varphi_w = \varphi_m \cos(\theta + \frac{2}{3}\pi)$$

Also, the induced voltages of the three phases are expressed by the following formulas, by using the above formula (1), when the angle speed is considered as ω.

$$E_u = \frac{d}{dt}\varphi_u = \frac{d}{dt}\varphi_m \cos\theta = -\omega\varphi_m \sin\theta = \omega\varphi_m \cos(\theta + \frac{\pi}{2})$$
$$E_v = \frac{d}{dt}\varphi_v = \frac{d}{dt}\varphi_m \cos(\theta - \frac{2}{3}\pi) = -\omega\varphi_m \sin(\theta - \frac{2}{3}\pi) = \omega\varphi_m \cos(\theta - \frac{\pi}{6})$$
$$E_w \frac{d}{dt}\varphi_w = \frac{d}{dt}\varphi_m \cos(\theta + \frac{2}{3}\pi) = -\omega\varphi_m \sin(\theta + \frac{2}{3}\pi) = \omega\varphi_m \cos(\theta + \frac{\pi}{6})$$

These formulas show that the induced voltage leads of π/2 phase from the permanent magnetic flux. This means that if the induced voltage can be detected, position of the permanent magnet can be estimated.

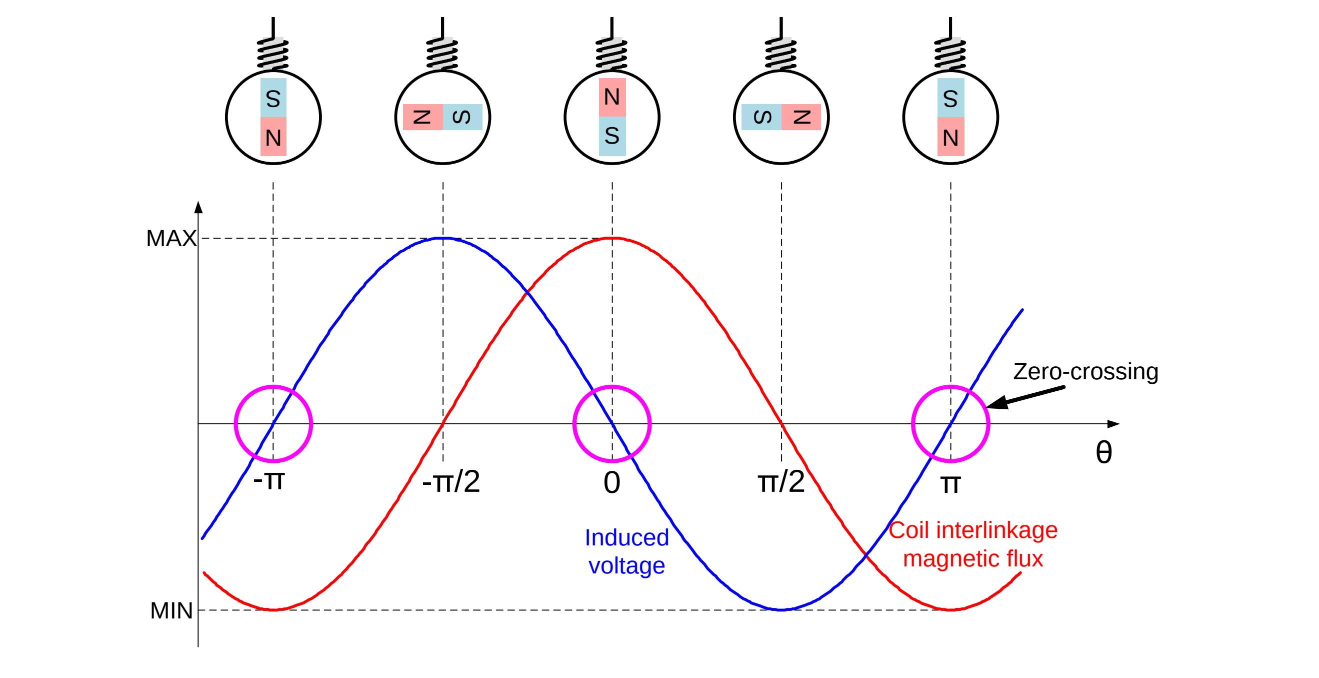


Figure 6-8 Zero-crossing of the induced voltage

However, the induced voltage of each phase may not be always detected while the motor is rotating.

While driving in 120-degree conduction, conduction is performed to two phases among the three. Therefore, in only the remaining one phase, to which current is not injected, the induced voltage can be detected. Actually, position information is obtained by detecting a change point of sign of induced voltage (zero-crossing) occurring in nonconducting phase.

In a three-phase motor, this zero-crossing occurs for total six times, i.e. twice in each phase, in one rotation (electrical angle) of the motor. This means that the position for every 60 degrees can be detected by this process with resolution equivalent to Hall sensors.

In this system, every time PWM control is performed, a pseudo motor center point voltage is obtained through A/D conversion of each phase voltage. By comparing the pseudo motor center point voltage with each phase voltage, the patterns of ‘1’ and ‘0’ are created according to the positional relation.

In addition, the pseudo Hall sensor pattern is created by shifting this created pattern by π/6 phase.

π/6 is a value calculated from the estimated rotational speed.

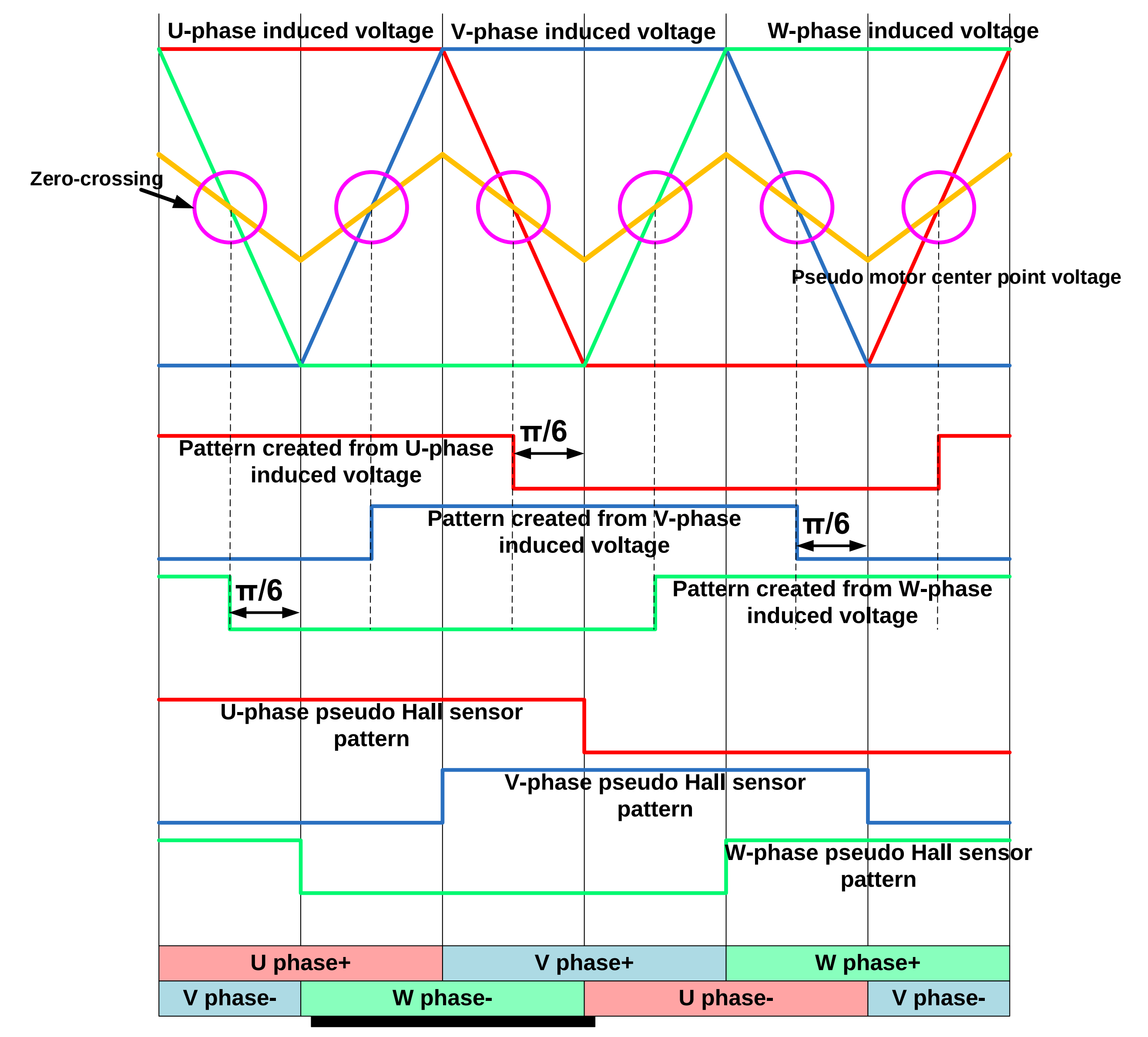


Figure 6-9 Pseudo Hall sensor pattern (In case of upper arm chopping)

Next, one of several methods to detect the zero-cross is described below. This method detects the zero-cross by using A/D converters in a microcontroller and comparing the induced voltage and center point voltage values by software. Since there is no need for a comparator to compare voltages, this method is also called “comparator-less method”.

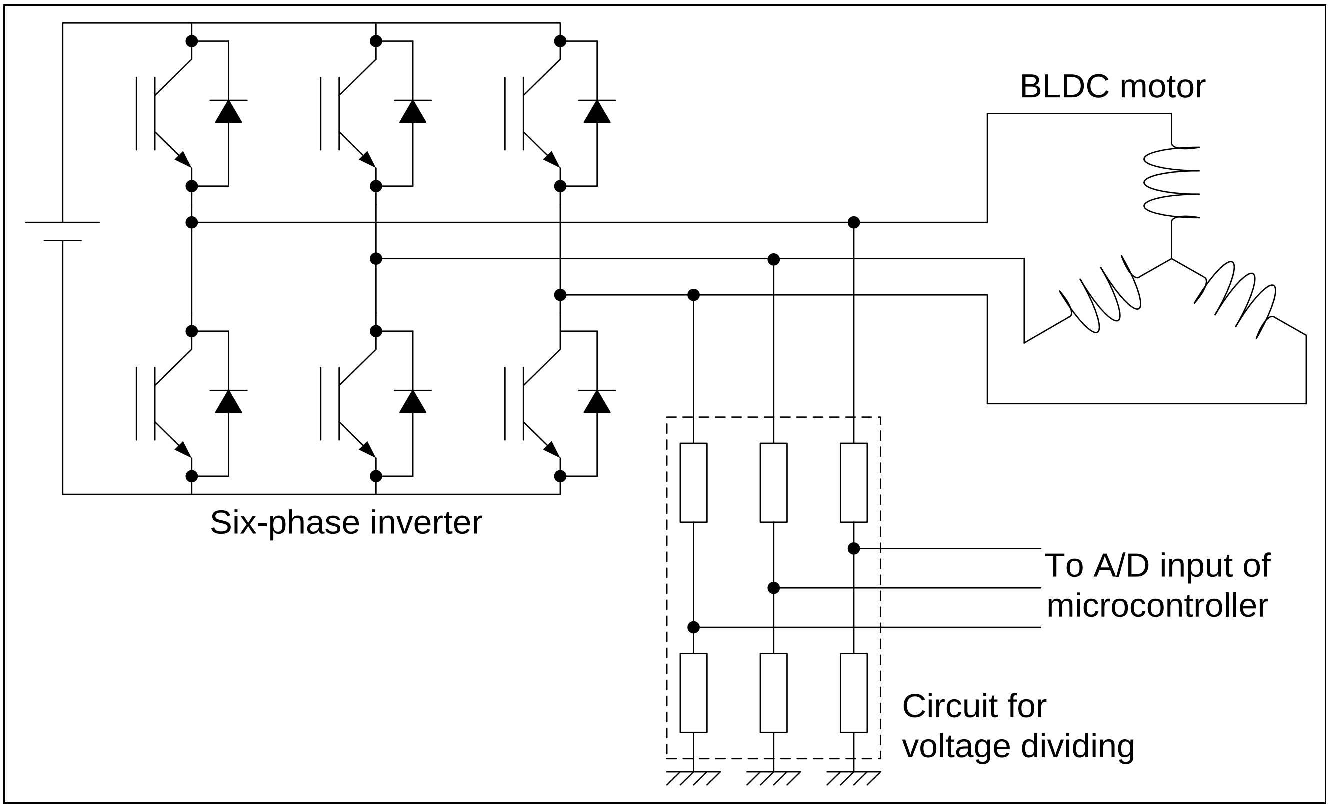


Figure 6-10 Comparator-less method

As for induced voltage to be detected actually, impact of commutation voltage generated when switching conducting patterns and PWM of other phases must be considered. The impact is expressed as shown in Figure 6-11. To reduce the impact, some countermeasures such as a method using a simple filter circuit or software filtering can be taken.

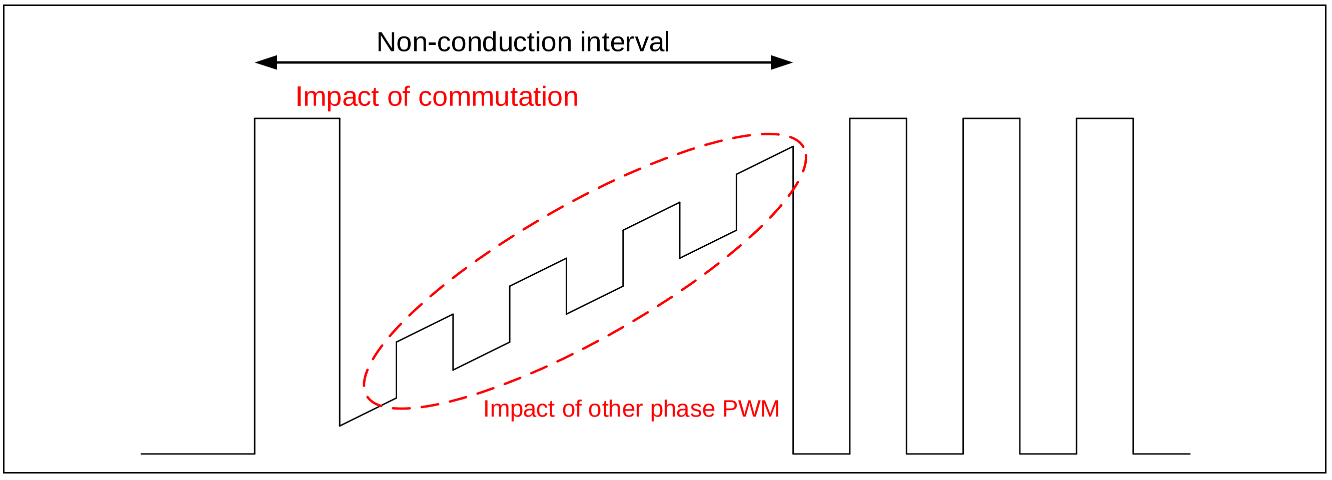


Figure 6-11 Conceptual Diagram of Impact of Commutation and Other Phase PWM

(b) Speed calculation

The motor rotational speed is calculated from a difference between the timer value confirmed 2π [rad] before and the current timer value. The timer values are obtained from a free-running timer peripheral of a microcontroller at the zero-cross point in which the conductive pattern change.

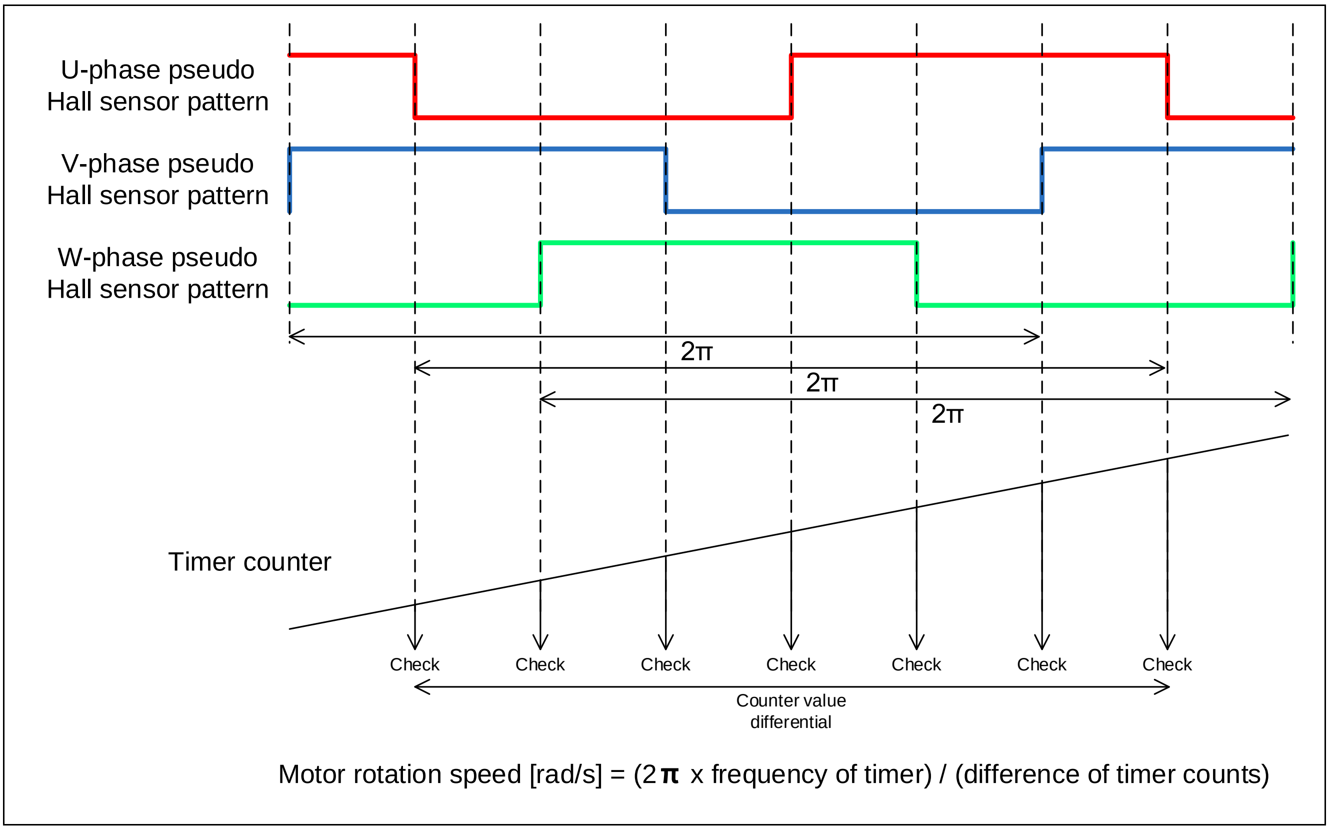


Figure 6-12 Method of calculation for the rotational speed

7. Test results

The test results provided in this chapter are for reference only and are the measurements taken in an environment as described in 2.1Test environments.

7.1 Program size

Table 7-1 shows 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 (Hall sensor)

Memory	RX13T	RX23T	RX24T	RX24U
ROM	16.3 KB	15.8 KB	16.2 KB	16.5 KB
RAM	4.9 KB	4.9 KB	5.0 KB	5.0 KB
Maximum value of stack analysis result	164 B	164 B	164 B	164 B
Stack size setting value	1536 B	1536 B	1536 B	1536 B

Table 7-2 Program size (Sensorless)

Memory	RX13T	RX23T	RX24T	RX24U
ROM	17.7 KB	17.2 KB	17.8 KB	17.6 KB
RAM	5.1 KB	5.1 KB	5.2 KB	5.2 KB
Maximum value of stack analysis result	204 B	204 B	204 B	204 B
Stack size setting value	1536 B	1536 B	1536 B	1536 B

7.2 CPU loading rate

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

Table 7-3 Control loop and CPU loading rate (Hall sensor)

MCU	Control loop type	Control interval	Processing time	CPU loading rate
RX24U	Carrier wave interrupt	50 µs (0 decimation)	8.7 µs	17.4 %
	Speed control loop	1000 µs	5.0 µs	0.5 %
RX24T	Carrier wave interrupt	50 µs (0 decimation)	8.4 µs	16.8 %
	Speed control loop	1000 µs	4.9 µs	0.5 %
RX23T	Carrier wave interrupt	50 µs (0 decimation)	16.0 µs	32.0 %
	Speed control loop	1000 µs	9.8 µs	1.0 %
RX13T	Carrier wave interrupt	50 µs (0 decimation)	19.2 µs	38.4 %
	Speed control loop	1000 µs	12.0 µs	1.2 %

Table 7-4 Control loop and CPU loading rate (Sensorless)

MCU	Control loop type	Control interval	Processing time	CPU loading rate
RX24U	Carrier wave interrupt	50 µs (0 decimation)	15.3 µs	30.6 %
	Speed control loop	1000 µs	4.1 µs	0.4 %
RX24T	Carrier wave interrupt	50 µs (0 decimation)	15.7 µs	31.4 %
	Speed control loop	1000 µs	4.1 µs	0.4 %
RX23T	Carrier wave interrupt	50 µs (0 decimation)	27.6 µs	55.2 %
	Speed control loop	1000 µs	8.8 µs	0.9 %
RX13T	Carrier wave interrupt	50 µs (0 decimation)	34.4 µs	68.8 %
	Speed control loop	1000 µs	10.4 µs	1.0 %

7.3 Operation waveforms

For your reference, a waveform that will be seen during control using RX24T is provided as the results of tests using sensors.

Table 7-5 Measurement conditions

Item	Value	Remarks
Kp, a control parameter for the speed control system	0.02f	
Ki, a control parameter for the speed control system	0.0005f 0.004f	Hall Sensorless
Load	—	Conducted at no load.

Figure 7-1 shows the results of testing speed control in a configuration with a Hall sensor.

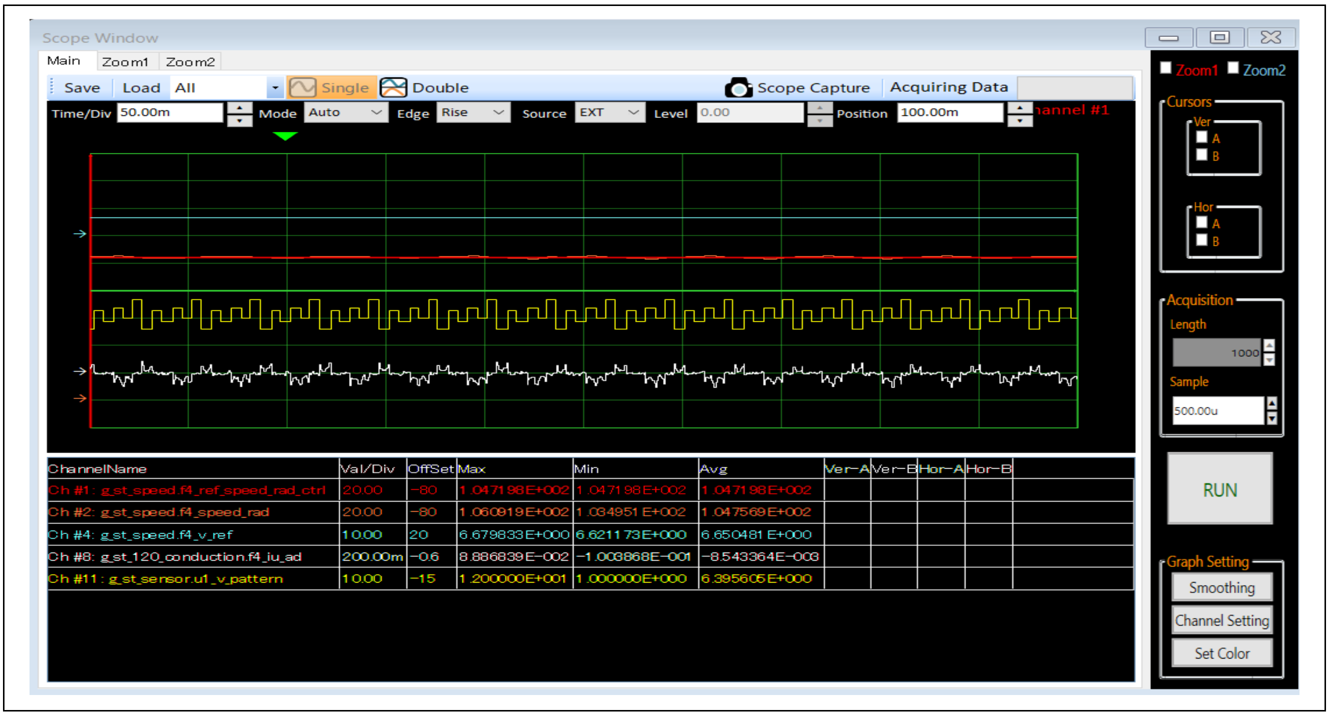


Figure 7-1 Speed control using Hall 120-degree conducting control

Drive condition:

Rotation speed: Speed command 1000 [rpm]

Waveform information:

Red: Instructed speed [rad/s] (20 rad/s/div.)
Orange: Detected speed [rad/s] (20 rad/s/div.)
Light blue: Instructed voltage [V] (10 V/Div)
Yellow: Conduction pattern
White: U-phase current [A] (200 mA/div.)
Horizontal axis: 50 ms/div.

Figure 7-2 shows the results of testing speed control in a sensorless drive configuration.

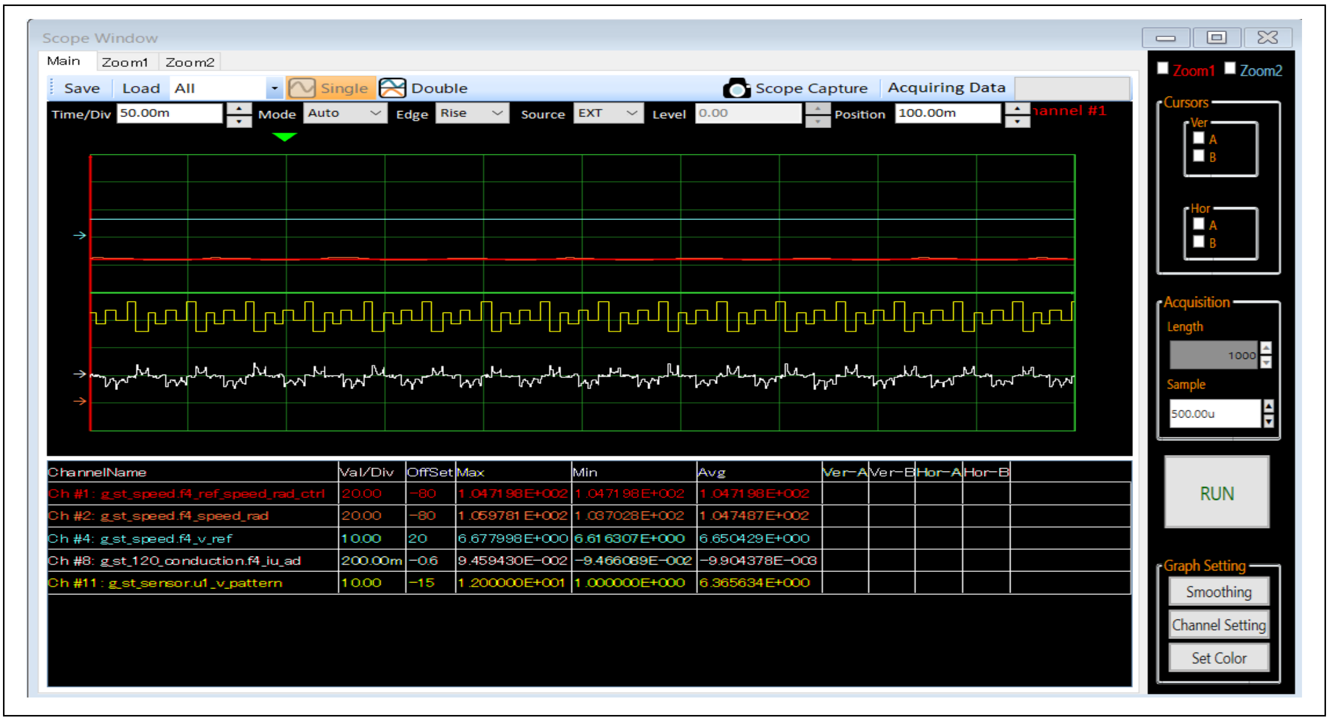


Figure 7-2 Speed control using sensorless 120-degree conducting control

Drive condition:

Rotation speed: Speed command 1000 [rpm]

Waveform information:

Red: Instructed speed [rad/s] (20 rad/s/div.)
Orange: Detected speed [rad/s] (20 rad/s/div.)
Light blue: Instructed voltage [V] (10 V/Div)
Yellow: Conduction pattern
White: U-phase current [A] (200 mA/div.)
Horizontal axis: 50 ms/div.

8. Reference materials

- Renesas Motor Workbench V.3.00 User's Guide (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)
- 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)

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/.