R01AN3991EJ0200
Rev.2.00
2020.10.20

RL78/G1F

Sensorless vector control for permanent magnetic synchronous motor

Summary

This application note explains the sample programs for driving a permanent magnet synchronous motor in the sensorless vector method using the RL78/G1F microcontroller. This note also explains how to use the motor control development support tool Renesas Motor Workbench (RMW).

These sample programs are intended to be used as references only, and Renesas Electronics Corporation does not guarantee their operation. Please use them after carrying out a thorough evaluation in a suitable environment.

Operation checking device

Operations of the sample programs have been checked using the following device.

- RL78/G1F(R5F11BLEAFB)

Applicable sample programs

This application note regards the following sample programs.

- RL78G1F_MRSSK_LESS_FOC_CSP_CC_V200 (IDE: CS+ for CC)
- RL78G1F_MRSSK_LESS_FOC_E2S_CC_V200 (IDE: e^{2}studio)

For 24V Motor Control Evaluation System for RX23T and RL78/G1F CPU card
RL78/G1F sensorless vector control sample program

References

- RL78/G1F Group User's Manual: Hardware (R01UH0516EJ0110)
- Sensorless vector control of permanent magnet synchronous motor: algorithm (R01AN3786JJ0101)
- Renesas Motor Workbench 2.0 User's Manual (R21UZ0004JJ0202: Renesas-Motor-Workbench-V2-0d)
- Renesas Solution Starter Kit 24V Motor Control Evaluation System for RX23T (Motor RSSK) User's Manual (R20UT3697JJ0100)
- RL78/G1F CPU Card User's Manual (R12UZ0014EJ0100)

Contents

1. Overview

This application note explains how to implement the sensorless vector control sample programs of the permanent magnetic synchronous motor (PMSM) using the RL78/G1F microcontroller, and how to use the motor control development support tool Renesas Motor Workbench. Note that these sample programs use part of the algorithm described in the application note “Sensorless vector control of permanent magnetic synchronous motor: algorithm.”

1.1 Development environment

Table 1-1 and Table 1-2 show the development environment of the sample programs explained in this application note.

Table 1-1 Development Environment of the Sample Programs (Hardware)

Microcontroller	Evaluation board	Motor
RL78/G1F (R5F11BLEAFB)	24V inverter board(Note 1) & RL78/G1F CPU card(Note 2)	TSUKASA(Note 3) TG-55L

Table 1-2 Development Environment of the Sample Programs (Software)

CS+ version	Build tool version
V8.03.00	CC-RL V1.09.00

e^{2}studio version	Build tool version
2020-07	CC-RL V1.09.00

For purchasing information and technical support, please contact Renesas Electronics Corporation sales representatives and dealers.

Notes:
1. The 24Vinverter board (RTK0EM0001B00012BJ) is a product of Renesas Electronics Corporation.
2. The following two kinds of RL78/G1F CPU Card can be used.
   RTK0EML240C03000BJ: Renesas Electronics Corporation
   T5103: Desk Top Laboratories Inc. (http://desktoplab.co.jp/)
3. TG-55L is a product of TSUKASA ELECTRIC.
   TSUKASA ELECTRIC. (https://www.tsukasa-d.co.jp/en/)

2. System overview

An overview of this system is provided below.

2.1 Hardware configuration

The hardware configuration is shown below.

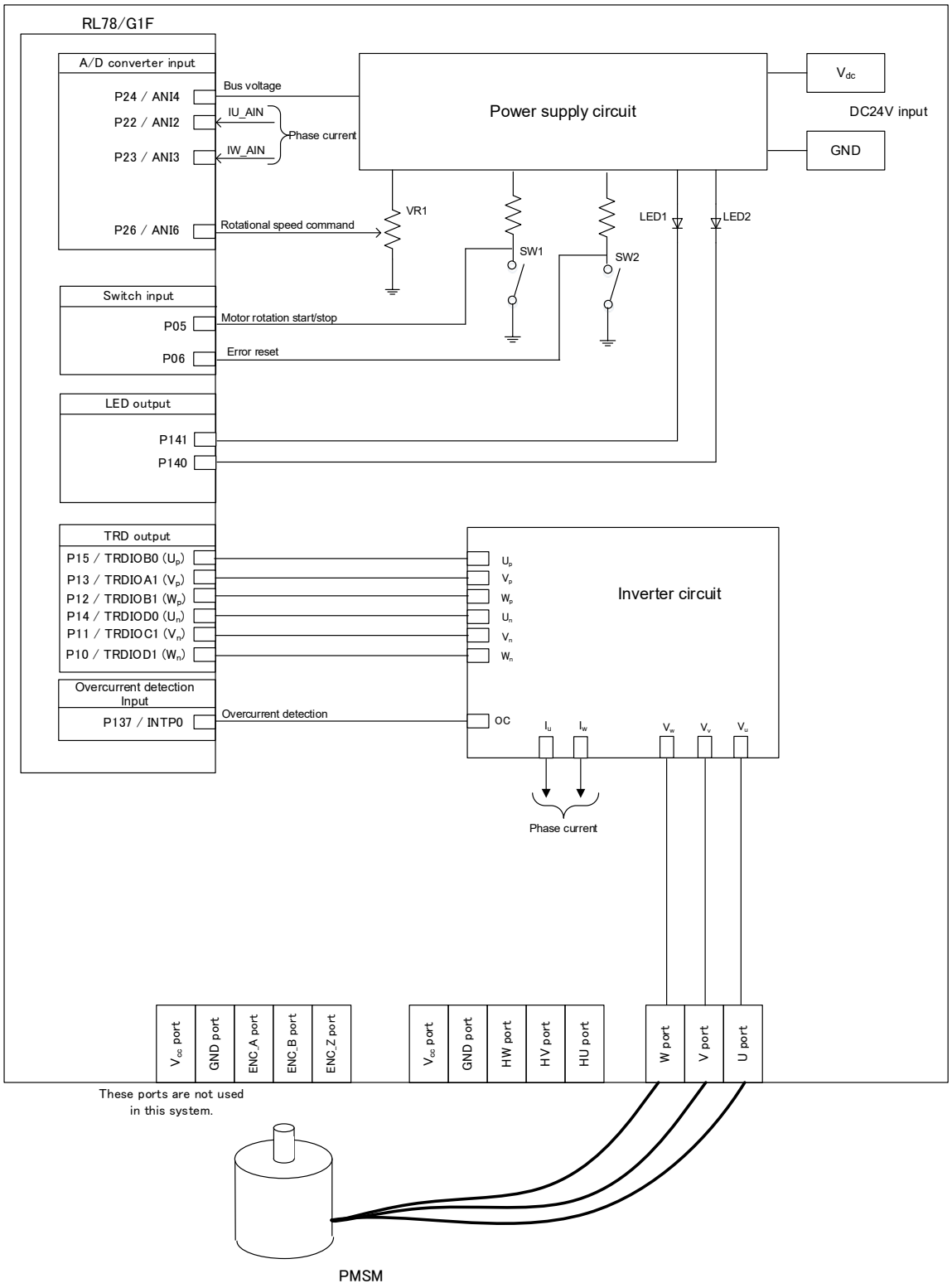

Figure 2-1 Hardware Configuration Diagram

2.2 Hardware specifications

2.2.1 User interface

Table 2-1 is a list of this system's user interface elements.

Table 2-1 User Interface

Item	Interface element	Function
VR1	Variable resistor	Rotational speed command value input (analog values)
SW1	Toggle switch	Motor rotation start/stop command
SW2	Toggle switch	Command of recovery from error status
LED1	Yellow-green LED	• During motor rotation: ON • When stopped: OFF
LED2	Yellow-green LED	• During error detection: ON • During normal operation: OFF
LED3	Yellow-green LED	Not used in this system
RESET	Push switch	System reset

The system's RL78/G1F microcontroller port interface elements are listed in Table 2-2.

Table 2-2 Port Interface

R5F11BLEAFB Port name	Function
P24 / ANI4	Inverter bus voltage measurement
P26 / ANI6	For inputting rotational speed command values (analog values)
P05	START / STOP toggle switch
P06	ERROR / RESET toggle switch
P141	LED1 ON / OFF control
P140	LED2 ON / OFF control
P04	LED3 ON / OFF control (not used in this system)
P22 / ANI2	U-phase current detection
P23 / ANI3	W-phase current detection
P15 / TRDIOB0	PWM output (U_p)
P13 / TRDIOA1	PWM output (V_p)
P12 / TRDIOB1	PWM output (W_p)
P14 / TRDIOD0	PWM output (U_n)
P11 / TRDIOC1	PWM output (V_n)
P10 / TRDIOD1	PWM output (W)
P137 / INTP0	PWM emergency stop input when overcurrent detected (external detection circuit used)

2.2.2 Peripheral functions

Table 2-3 is a list of peripheral functions used in this system.

Table 2-3 List of Peripheral Functions

Peripheral Function	Usage
10-bit A/D converter (AD)	• Rotational speed command value input • Inverter bus voltage measurement • U and W phase current measurement (V phase current is calculated from U and W phase currents)
Timer Array Unit (TAU)	• 1-ms interval timer • Control cycle timer
Timer RD (TRD)	PWM output for complementary PWM mode use
External interrupt (INTP0)	Overcurrent detection

(1) 10-bit A/D converter (AD)

The rotational speed command value input, U phase current (Iu), W phase current (Iw), and inverter bus voltage (Vdc) are measured using the 10-bit A/D converter.

A/D conversion sets the channel selection mode to Select mode and the conversion operation mode to One-shot Conversion mode. (Uses a software trigger).

(2) Timer Array Unit (TAU)

a. 1-ms interval timer

Uses channel 0 of the Timer Array Unit as the 1-ms interval timer.

b. Control cycle timer

Uses channel 1 of the Timer Array Unit.
It is used as a 100-µs interval timer.

(3) Timer RD (TRD)

Using complementary PWM mode, outputs six-phase PWM with deadtime.

Also, using a pulse output forced blocking function, the PWM output port outputs high impedance when overcurrent is detected.

(4) External interrupt (INTP0)

Overcurrent is detected using an external circuit (using a pulse output forced blocking function).

2.3 Software structure

2.3.1 Software file structure

The folder and file configurations of the sample programs are given in Table 2-4 below.

Table 2-4 Folder and File Configurations of the Sample Programs

Folder		File	Content
config		r_mtr_config.h	Configuration definition
		r_mtr_motor_parameter.h	Motor parameter definition
		r_mtr_control_parameter.h	Control parameter definition
		r_mtr_inverter_parameter.h	Inverter parameter definition
		r_mtr_scaling_parameter.h	Scaling parameter definition
application	main	main.h main.c	Main function
	board	r_mtr_board.h r_mtr_board.c	Function definition for hardwareUI
	ics	r_mtr_ics.h r_mtr_ics.c	Function definition for Analyzer(Note1) UI
		ICS_define.h	CPU definition for RMW
		RL78G1F_vector.c	Interrupt vector function definition for RMW
		Ics2_RL78G1F.h	Function declaration for RMW communication
		ics_RL78G1F_Lx.h	MCU serial communication definition for RMW communication
		ICS2_RL78G1F.obj	Library for RMW communication
driver	auto_generation	cstart.asm hdwinit.asm stkinit.asm iodefine.h	Auto generation files
		r_mtr_rl78g1f.h r_mtr_rl78g1f.c	Function definition for MCU control
middle		r_dsp_cc.h R_dsp_rl78_CC.lib	DSP definition
		r_mtr_common.h	Common definition
		r_mtr_parameter.h	Motor control parameter definition
		r_mtr_sc_table.h	Trinodal function table
		r_mtr_ctrl_gain.obj	Gain design function definition
		r_mtr_driver_access.h r_mtr_driver_access.c	Function definition for driver access
		r_mtr_statemachine.h r_mtr_statemachine.c	Function definition for state machine
		r_mtr_foc_less_speed.h r_mtr_foc_less_speed.c	Sensorless vector control-related function definition
		r_mtr_interrupt.c	Interrupt handler function definition
		r_mtr_est_phase_err.h r_mtr_est_phase_err.obj	Axis error estimating function definition

Note 1: Regarding the specification of the Analyzer function in the motor control development support tool Renesas Motor Workbench (RMW), please refer to Chapter 4. The identifier ics/ICS (ICS is the previous motor control development support tool, In Circuit Scope) is attached to the names of folders, files, functions, and variables related to Renesas Motor Workbench.

2.3.2 Module configuration

Figure 2-2 shows the module configuration of the sample programs.

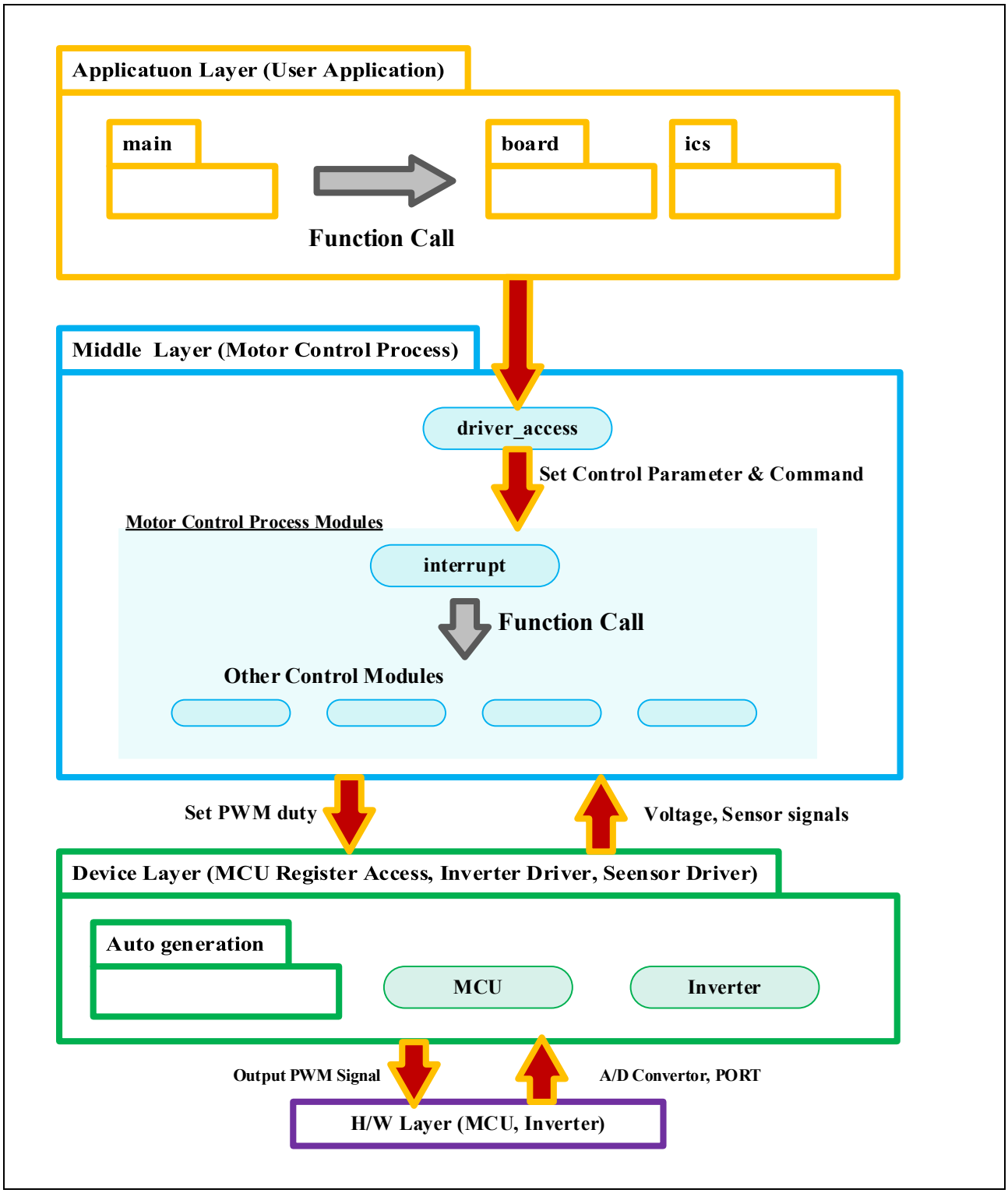

Figure 2-2 Module Configuration of the Sample Programs

2.4 Software specifications

The basic software specifications of the system are listed below.

Table 2-5 Software Specifications

Item	Content	
Control method	Vector control	
Motor rotation start/stop	Determined depending on the level of SW1 (P05) (“Low”: rotation start, “High”: stop) Or, operation using the motor control development support tool[Note]	
Position detection of rotor magnetic pole	Sensorless	
Input voltage	DC 24 V	
Main clock frequency	CPU clock: f_{CLK} 32 MHz TRD clock: f_{HOCO} 64 MHz	
Carrier frequency (PWM)	20 kHz	
Deadtime	2 µs	
Control cycle	Current control location/speed estimate: 100 µs (twice the carrier cycle) Speed control: 1 ms	
Rotational speed range	CW: 0 rpm - 2650 rpm CCW: 0 rpm - 2650 rpm However, driving is performed as an open loop at 1060 rpm or less	
Optimal setting	Default setting	
ROM/RAM Size	ROM	11.6 KB
	RAM	0.85 KB
Processing stop for protection	- Disables the motor control signal output (six outputs), under any of the following conditions. 1. Inverter bus voltage exceeds 28 V 2. Inverter bus voltage falls below 15 V 3. Rotational speed exceeds 3975 rpm 4. Each current phase exceeds 2.0 A - When an overcurrent detection signal from the outside (low level input to INTP0 port) is detected, the PWM output ports are made high impedance.	

[Note] For details, see “Usage of Motor Control Development Support Tool, Renesas Motor Workbench.”

2.5 User option bytes

The settings of the user option byte area of the RL78/G1F flash memory are shown below.

Table 2-6 User option byte settings

Setting	Address	Value	Description
787BF8	000C0H /010C0H	01111000B	- Uses watchdog timer interval interrupt: does not use interval interrupt - Period when watchdog timer window is open: 100% - Watchdog timer counter operation control: Counter operation possible (After reset is canceled, count begins) - Watchdog timer overflow time: 136 ms - Watchdog timer counter operation control: In HALT/STOP mode, counter operation stops
	000C1H /010C1H	01110011B	- LVD settings (reset mode) Rising edge: 2.92 V Falling edge: 2.86 V
	000C2H /010C2H	11111000B	- Flash operation mode setting: HS (high-speed main) mode - High-speed on-chip oscillator/block frequency fHOCO: 64 MHz fIH: 32 MHz

3. Explanation of Control Programs

The sample programs to which this application note applies are explained here.

3.1 Contents of control

3.1.1 Motor start/stop

Starting and stopping of the motor are controlled by input from Renesas Motor Workbench or SW1 & VR1. A general-purpose port is assigned to SW1. The port is read within the main loop. When the port is at a “Low” level, it is determined that the start switch is being pressed. Conversely, when the level is switched to “High”, the program determines that the motor should be stopped.

3.1.2 A/D converter

(1) Motor rotational speed command value

The motor rotational speed command value can be set by Renesas Motor Workbench input or A/D conversion of the VR1 output value (analog value). The A/D converted VR1 value is used as the rotational speed command value, as shown below. Here, the maximum value of the command value is set as the maximum rpm that can be obtained by the resolution of the A/D converter.

Table 3-1 Conversion Ratio of the Rotational Speed Command Value

Item	Conversion ratio (Command value: A/D conversion value)		Channel
Rotational speed command value	CW	0 rpm – 2650 rpm: 0200H – 03FFH	ANI6
	CCW	-2650 rpm – 0 rpm: 0000H – 01FFH	

(2) Inverter bus voltage

The inverter bus voltage is measured as given in Table 3-2. It is used for modulation factor calculation and over- and undervoltage detection. (When an abnormality is detected, PWM is stopped).

Table 3-2 Inverter Bus Voltage Conversion Ratio

Item	Conversion ratio (Inverter bus voltage: A/D conversion value)	Channel
Inverter bus voltage	0 V – 111 V: 0000H – 03FFH	ANI4

(3) U phase and W phase current

As shown in the table below, U phase and W phase current are measured and used for vector control.

Table 3-3 Conversion Ratios of U and W Phase Currents

Item	Conversion ratio (U phase and W phase currents: A/D conversion value)	Channel
U phase and W phase current	-10 A – 10 A: 0000H – 03FFH	Iu: ANI2 Iw: ANI3

[Note] For more information about A/D conversion characteristics, see “RL78/G1F User’s Manual - Hardware.”

3.1.3 Voltage control by PWM

PWM control is used for controlling output voltage. PWM control is a control method that continuously adjusts the average voltage by varying the pulse duty, as shown in Figure 3-1.

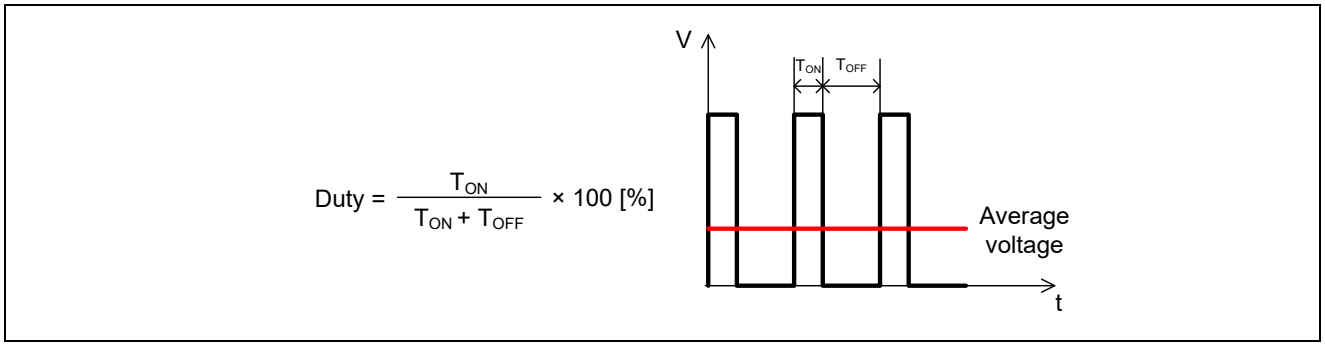

Figure 3-1 PWM Control

Here, the modulation factor *m* is defined as follows.

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

m: Modulation factor V: Command value voltage E: Inverter bus voltage

3.1.4 Modulation

The input voltage to the motor is signal generated by pulse-width modulation (below, PWM) and applied. This section explains the method of creating the PWM pulse width.

(1) Triangle Wave Comparison Method

The triangle wave comparison method is one method for actually outputting command value voltage. The pulse width of the output voltage is determined by comparing the carrier waveform (triangle wave) and the command value voltage waveform. A sine wave-shaped command value voltage can be output artificially by turning the switch on when the command value voltage is greater than the carrier wave voltage, and turning the switch off when it is smaller.

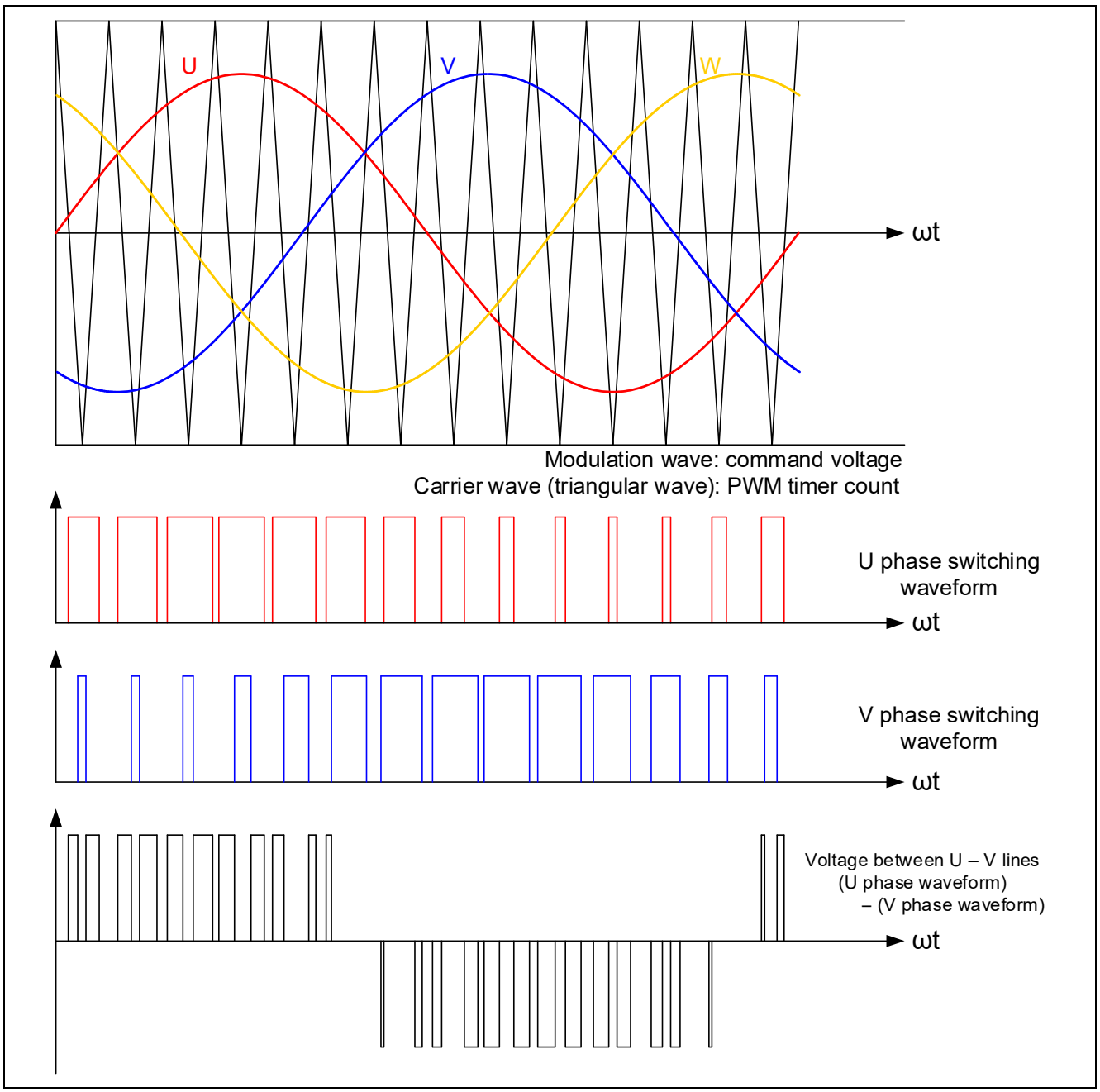

Figure 3-2 Conceptual Diagram of the Triangle Wave Comparison Method

(2) Third Harmonic Injection Method

In the triangle wave comparison method, only approximately 86.6% of the direct current voltage at which the line voltage amplitude is input can be used. There are many modulation methods for improving voltage utilization efficiency, but in this program the third harmonic imposition method can be used. By calculating the command voltage as shown below, the command voltage becomes the same as with third harmonic waves imposed.

$$v_o = \frac{\max(v_u^*, v_v^*, v_w^*) + \min(v_u^*, v_v^*, v_w^*)}{2}$$

$$\overline{v_u^*} = v_u^* - v_o$$

$$\overline{v_v^*} = v_v^* - v_o$$

$$\overline{v_w^*} = v_w^* - v_o$$

v_u^*, v_v^*, v_w^* : Original UVW phase command voltage

$\overline{v_u^*}, \overline{v_v^*}, \overline{v_w^*}$: UVW phase command voltage of 3rd harmonic superimposition method

v_o : Resistance

It is possible to change the above modulation method by setting the following values to MOD_METHOD in r_mtr_config.h, and compiling it.

MOD_3PH_SPWM	Triangle Wave Comparison Method	0
MOD_3PH_TOW	Third Harmonic Imposition Method	1: Default setting

3.1.5 State transitions

The state transition diagram for this program is shown in Figure 3-3.

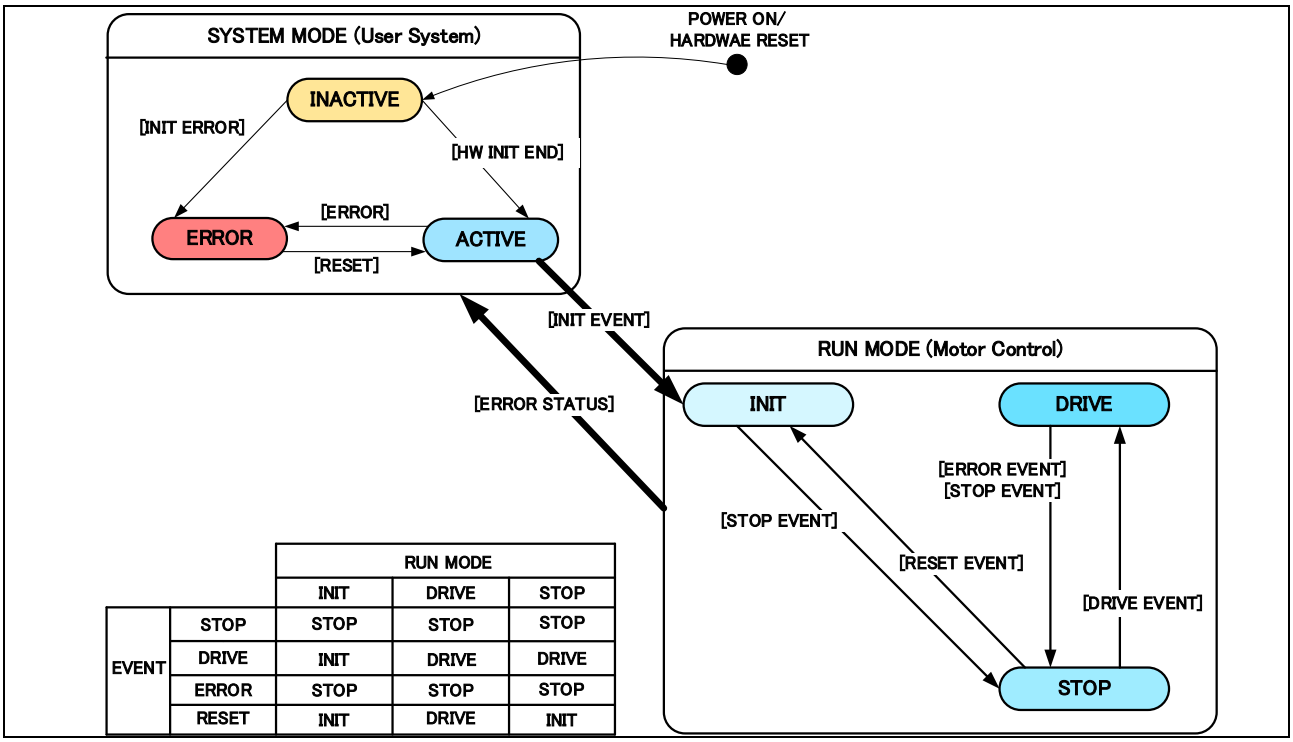

Figure 3-3 State Transition Diagram

(1) SYSTEM MODE

SYSTEM MODE indicates the operating state of the system. SYSTEM MODE has three states, which are the motor drive stop (INACTIVE), motor drive (ACTIVE), and abnormal condition (ERROR) states.

(2) RUN MODE

RUN MODE indicates the drive condition of the motor. The state is changed by the occurrence of an EVENT.

(3) EVENT

EVENT indicates a change in RUN MODE. When an EVENT occurs, the RUN MODE changes as shown in Figure 3-3. Each EVENT is caused by an occurrence as shown in Table 3-4.

Table 3-4 EVENT List

EVENT name	Occurrence factor
STOP	By user operation
DRIVE	By user operation
ERROR	When the system detects an error
RESET	By user operation

3.1.6 Startup method

The description of startup control of the sensorless vector control software is shown in Figure 3-4. The mode is controlled by the states that control the command values of the d-axis current, q-axis current, and speed.

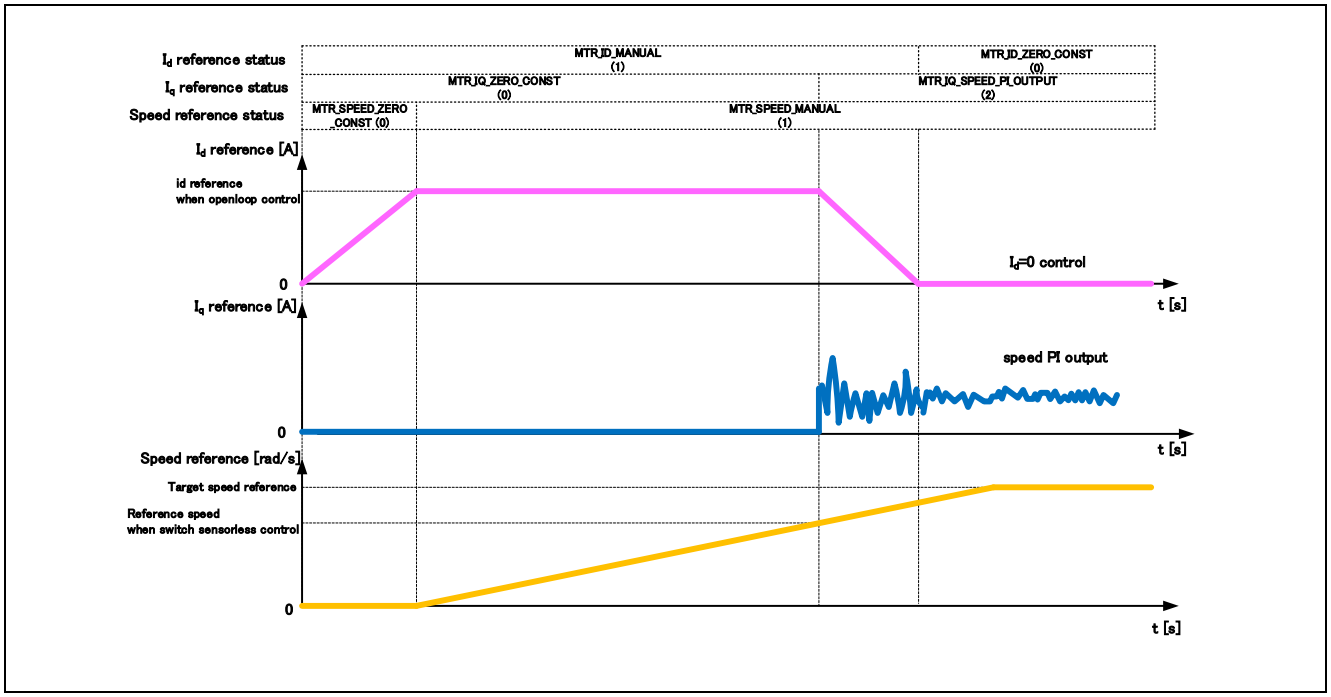

Figure 3-4 Description of Sensorless Speed Control Software Startup Control

3.1.7 Control method

The block diagram of the entire control system is shown in Figure 3-5. The control system is made up of a coordinate converter and decoupling controller, axis error estimator, PLL controller, Auto Speed Regulator (ASR), and Auto Current Regulator (ACR). Also, a primary LPF is set up for the estimated speed and δ-axis current in order to prevent axis error estimate pulsations, etc. due to factors such as disturbances.

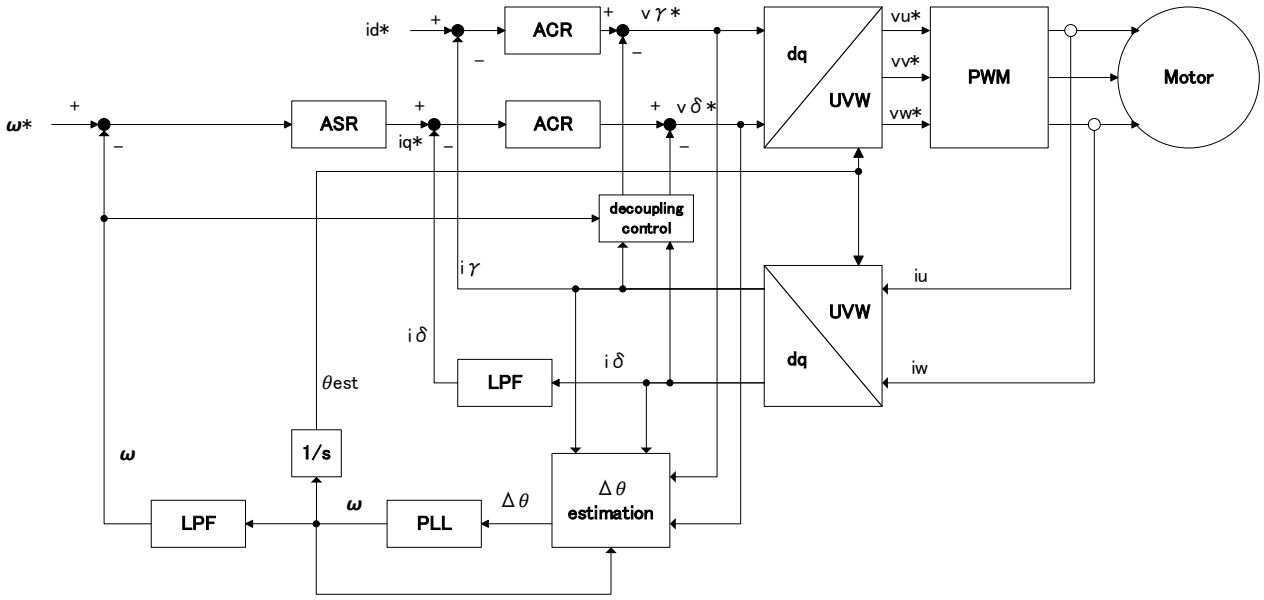

Figure 3-5 Control System Block Diagram

The axis error estimator estimates the axis error $\Delta\theta$ between the actual dq axis and the estimated $\gamma\delta$ axis. The voltage equation for the $\gamma\delta$ axis can be obtained by multiplying the rotation matrix in the formula shown below to both sides of the voltage equation for the dq axis in the following equation.

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

v_d, v_q: d-axis voltage, q-axis voltage
i_d, i_q: d-axis current, q-axis current
R_a: resistance

ω: angular speed
L_d, L_q: dq axis inductance
$I?_a$: Effective value of electronic interconnected magnetic flux due to permanent magnet

The following formula is the rotation matrix with which the above equation is multiplied.

$$\begin{bmatrix} cos\Delta\theta & -sin\Delta\theta \\ sin\Delta\theta & cos\Delta\theta \end{bmatrix}$$

The induced voltage constituent elements e_γ $e_{I3}$and e_δ of the γ-axis and δ-axis are calculated, and the axis error $\Delta\theta$ is obtained using the following equation. When calculating the induced voltage, the γ-axis voltage and the δ-axis voltage are each used in approximation with voltage command values v_γ^* and v_δ^*. The estimated speed ω is found by constructing a feedback loop so that this axis error reaches 0 (PLL controller).

$$\Delta\theta = \mathrm{atan}\left(\frac{e_\gamma}{e_\delta}\right)$$

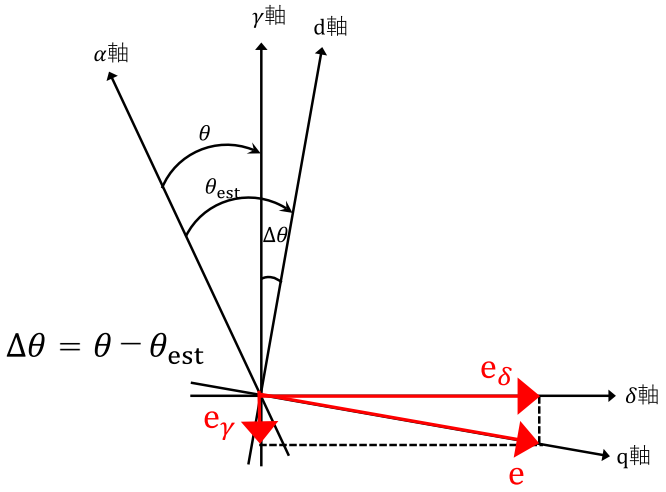

Figure 3-6 Axis Error and γ- and δ-axis Induced Voltage Constituent Elements

The ACR, ASR, and PLL controller are achieved by using the PI controller. Their gain requires suitable adjustment combined with the desired controls. The current PI control gain K_{pACR} and $K_{i_{ACR}}$, the speed PI control gain K_{pACR} and $K_{i_{ACR}}$, and the PLL control gain K_{pPLL} and $K_{i_{PLL}}$ are each defined as in the following formulas.

$$K_{p_{ACR}} = \omega_{CG} L, \qquad K_{i_{ACR}} = \omega_{CG} R$$

ω_{CG}: Current PI control natural frequency

L: Inductance (L_d at d-axis, L_q at q-axis)

R: Resistance

$$K_{p_{ASR}} = \frac{\omega_{SG} J}{P_n^2 \psi_a}, \qquad K_{i_{ASR}} = \frac{\omega_{SG}^2 J}{5 P_n^2 \psi_a}$$

ω_{SG}: Speed PI control natural frequency

J: Inertia

ψ_a: Induced voltage coefficient

P_n: Number of pole pairs

$$K_{p_{PLL}} = \omega_{\Delta\theta}, \quad K_{i_{PLL}} = \frac{\omega_{\Delta\theta}^2}{5}$$

$\omega_{\Delta\theta}$: PLL control natural frequency

3.1.8 System protection function

This program has the following types of error states, and executes an emergency stop function in the event that any of the following errors occur. Refer to Table 3-5 for the settings of the system protection functions.

- Overcurrent error for hardware

When an emergency stop signal (overcurrent detection) from the external hardware is detected, voltage output is stopped.

- Overcurrent error

U phase, V phase, and W phase current are monitored in the overvoltage monitoring cycle. When overvoltage (value exceeding the overvoltage limit) is detected, an emergency stop occurs.

- Overvoltage error

The inverter bus voltage is monitored in the overvoltage monitoring cycle. When overvoltage (value exceeding the overvoltage limit) is detected, an emergency stop occurs. The overvoltage limit is set in consideration of the error of the resistance value of the detection circuit.

- Undervoltage error

The inverter bus voltage is monitored in the undervoltage monitoring cycle. When undervoltage is detected (when it goes below the undervoltage limit), an emergency stop occurs. The undervoltage limit is set in consideration of the error of the resistance value of the detection circuit.

- Rotational speed error

The speed is monitored in the rotational speed monitoring cycle. When the speed limit value is exceeded, an emergency stop occurs.

Table 3-5 System Protection Function Settings

Kinds of error	Threshold	
Overcurrent error for hardware	Overcurrent limit [A]	2.0
Overcurrent error	Overcurrent limit [A]	2.0
	Monitoring cycle [μs]	100
Overvoltage error	Overvoltage limit [V]	28
	Monitoring cycle [μs]	100
Undervoltage error	Undervoltage limit [V]	15
	Monitoring cycle [μs]	100
Rotational speed error	Speed limit [rpm]	3500
	Monitoring cycle [μs]	100

3.1.9 Per-unit method (PU)

The dynamic range of motor control is determined during compiling using fixed point arithmetic. If there is a large difference between the actual motor characteristic and the hypothetical motor characteristic during design, problems such as overflow and rounding errors tend to occur due to differences in dynamic ranges. The program uses the per-unit method (PU: per-unit) in order to reduce the calculated dynamic range's dependency on the motor characteristics. The PU value of any physical quantity is its value relative to a physical value serving as a standard, and can be derived as follows:

$$PU\ Value = \frac{Physical\ quantity}{Base\ Value}$$

All PU units used for control, such as physical quantity and gain, can be derived from the base current, base voltage, base frequency, and base angle. For example, base resistance can be calculated from the base voltage and base current:

$$Base\ Resistance = \frac{Base\ Voltage}{Base\ Current}$$

The effect of motor characteristics on calculated dynamic range is reduced, so it is necessary to set standard values for current, voltage, and angular frequency based on the motor characteristics (the method of deriving the standard value is not unique). In this program, rated current, voltage input to inverter, and maximum speed are set to standard values (PU units) for current, voltage, and angular frequency. The base value for each physical quantity is shown in Table 3-6. These values are defined in r_mtr_scaling_parameter.h.

Table 3-6 PU system base values

Category	Item	Definition	Unit
PU base physical quantity	Current	Rated current	[A]
	Voltage	Input voltage (inverter input)	[V]
	Angular frequency	2π × maximum speed [rpm] × number of pole pairs/60	[Hz]
	Angle	1	[rad]
Physical quantity	Time	Angle / Angular frequency	[s]
	Resistance	Voltage / Current	[Ω]
	Inductance	Resistance / Angular frequency	[H]
	Magnetic flux	Voltage / Angular frequency	[Wb]
	Inertia	Magnetic flux × current × (number of pole pairs / angular frequency)^2	[kgm^2/rad]
Current control	Kp	Resistance	[Ω]
	Kidt	Resistance	[Ω]
Speed control	Kp	Current / angular frequency	[A/(rad/s)]
	Kidt	Current / angular frequency	[A/(rad/s)]
PLL control	Kp	Angular frequency / angle	[Hz]
	Kidt	Angular frequency / angle	[Hz]

3.2 Sensorless Vector Control Software Function Specification

A list of functions used in this control program is provided below.

Table 3-7 List of Functions in “main.c”

File	Function	Process overview
main.c	main Input: none Output: none	- Call hardware initialization function - Call user interface initialization function - Call main processing use variable initialization function - Call state transition and event execution function - Call bus voltage stability waiting process - Main process ⇒Call user interface process ⇒Call watchdog timer clear function
	board_ui Input: none Output: none	Board user interface - Motor status change - Determination of rotational speed command value - Determination of direction of rotation
	ics_ui Input: none Output: none	Uses Renesas Motor Workbench - Motor status change - Rotational speed command value determination
	software_init Input: none Output: none	Initialization of variable used for main process

Table 3-8 List of Functions in “r_mtr_ics.c”

File	Function	Process overview
r_mtr_ics.c	mtr_set_com_variables Input: none Output: none	Preprocess to set control variables - input values of com variables to ICS variables - input values of ICS variables to ICS buffer variables
	mtr_ics_variables_init Input: none Output: none	Initialization of com variables
	R_MTR_Limit Input: int16_t s2_value / target value int16_t s2_max / maximum value (int16_t) s2_min / minimum limit Output: int16_t s2_temp / limited value	Limit between maximum and minimum values

Table 3-9 – List of Functions in “ics_RL78G1F.obj”

File name	Function name	Processing overview
main.c	ics2_init argument: unsigned int addr / DTC vector table start address char pin / Pins used by SCI char level / Interrupt level char num / Top address of DTC structure char brr / communication speed char mode / Communication mode return: none	Communication initialization
	ics2_watchpoint argument: none return: none	Call transfer function Must be called at intervals of 250us or more.

Table 3-10 List of Functions in “r_mtr_board.c”

File	Function	Process overview
r_mtr_board.c	mtr_board_led_control Input: (uint8_t u1_motor_status / motor status (uint8_t) u1_system_status / system status Output: none	LED control
	mtr_remove_chattering Input: (uint8_t) u1_sw / switch input signal (uint8_t) u1_on_off / switch status Output: (uint8_t) u1_flag_chattering / flag for chattering	Remove chattering of switch input signal

Table 3-11 List of Functions in “r_mtr_ctrl_rl78g1f.c”

File	Function	Process overview
mtr_ctrl_rl78g1f.c	R_MTR_InitHardware Input: none Output: none	Initialization of clock and peripheral functions
	R_MTR_CtrlStart Input: none Output: none	Timer RD (PWM) output authorization
	R_MTR_CtrlStop Input: none Output: none	Timer RD (PWM) output stopped Initialization of register
	R_MTR_GetAdc Input: uint8_t u1_ad_ch / A/D channel Output: uint16_t / A/D conversion result	A/D conversion
	R_MTR_GetIuwAdc Input: uint16_t *u2_ad_iuwvdc / UW phase pointer Output: none	UW phase current detection A/D conversion
	R_MTR_GetVdcAdc Input: none Output: none	Voltage detection A/D conversion
	R_MTR_ClearOcFlag Input: none Output: none	Clear overcurrent flag
	mtr_init_clock (inline function) Input: none Output: uint16_t / clock setting error	Initialization of clock
	mtr_init_ui (inline function) Input: none Output: none	Initialization of user interface
	mtr_init_tau (inline function) Input: none Output: none	Initialization of timer array unit (TAU)
	mtr_init_inttm00_interrupt (inline function) Input: none Output: none	TAU00 interrupt setting initialization
	mtr_init_inttm01_interrupt (inline function) Input: none Output: none	TAU01 interrupt setting initialization
	mtr_init_trd (inline function) Input: none Output: none	Clear watchdog timer (WDT)
	mtr_init_ad_converter (inline function) Input: none Output: none	Initialization of A/D converter
	mtr_init_pwm_register (inline function) Input: none Output: none	Initialization of PWM output register
	mtr_init_intp (inline function) Input: none Output: none	Cancel PWM high-impedance state

Table 3-12 List of Functions in “r_mtr_ctrl_gain.obj”

File	Function	Process overview
r_mtr_ctrl_gain.obj	mtr_ctrl_gain Input: st_mtr_foc_t *st_foc :: FOC structure pointer const st_mtr_design_parameter_t *st_ctrl_param :: design parameter structure pointer Output: none	Gain design process

Table 3-13 List of Functions in “r_mtr_driver_access.c”

File	Function	Process overview
r_mtr_driver_access.c	R_MTR_InitControl Input: none Output: none	Initialization of motor control system - initialization of motor status - initialization of control variables
	R_MTR_ExecEvent Input: uint8_t u1_event :: event Output: none	Change motor status and execute event process
	R_MTR_ChargeCapacitor Input: none Output: (uint16_t) u2_charge_cap_error :: timeout error	Waiting for stability of bus voltage
	R_MTR_SetSpeed Input: (int16_t) s2_ref_speed_rpm / target rotational speed Output: none	Set speed command value
	R_MTR_GetSpeed Input: none Output: int16_ts2_speed_rpm :: rotational speed	Get speed
	R_MTR_Get_Dir Input: none Output: (uint8_t) g_st_120.u1_dir :: direction of rotation	Get direction of rotation
	R_MTR_GetStatus Input: none Output: (uint8_t) mtr_statemachine_get_status(g_st_120.st_stm) :: motor status	Get motor status
	R_MTR_GetErrorStatus Input: none Output: (uint16_t) g_st_120.u2_error_status :: error status	Get error status
	R_MTR_IcsInput Input: (mtr_ctrl_input_t) *st_ics_input ::ICS structure Output: none	Input values of ICS variables to ICS buffer variables
	R_MTR_SetVariables (inline function) Input: none Output: none	Input values of ICS buffer variables to control variables
	R_MTR_InputBuffParamReset Input: none Output: none	Reset ICS buffer variables
	R_MTR_UpdatePolling Input: none Output: none	Set control variables

Table 3-14 List of Functions in “r_mtr_statemachine.c”

File	Function	Process overview
r_mtr_statemachine.c	mtr_statemachine_init Input: (st_mtr_statemachine_t) *p_state_machine :: motor status structure Output: none	Initialization of motor status
	mtr_statemacine_reset Input: (st_mtr_statemachine_t) *p_state_machine :: motor status structure Output: none	Reset motor status
	mtr_state_machine_event Input: (st_mtr_statemachine_t) *p_state_machine :: motor status structure (void) *p_object ::structure for control variables (uint8_t) u1_event ::event Output: none	Execute event
	mtr_statemachine_get_status Input: (st_mtr_statemachine_t) *p_state_machine :: motor status structure Output: (uint8_t) p_state_machine->u1_status ::motor status	Get motor status
	mtr_act_none Input: (st_mtr_statemachine_t) *st_stm :: motor status structure (void) *p_param ::structure for control variables Output: none	No process is performed
	mtr_act_init Input: (st_mtr_statemachine_t) *st_stm :: motor status structure (void) *p_param ::structure for control variables Output: none	Initialization of control variables
	mtr_act_error Input: (st_mtr_statemachine_t) *st_stm :: motor status structure (void) *p_param ::structure for control variables Output: none	Stop motor
	mtr_act_drive Input: (st_mtr_statemachine_t) *st_stm :: motor status structure (void) *p_param ::structure for control variables Output: none	Reset control variables
	mtr_act_stop Input: (st_mtr_statemachine_t) *st_stm :: motor status structure (void) *p_param ::structure for control variables Output: none	Stop motor

Table 3-15 List of Functions in “r_mtr_foc_less_speed.c”

File	Function	Process overview
r_mtr_foc_less_speed.c	mtr_foc_motor_default_init Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: none	Initialization of control variables
	mtr_foc_motor_reset Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: none	Reset control variables
	mtr_ctrl_gain_apply Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: none	Transfer from control gain buffer variable to control variable

Table 3-16 List of Functions in “r_mtr_est_phase_err.obj”

File	Function	Process overview
r_mtr_est_phase_err.obj	mtr_est_phase_err Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: none	Axis error estimating process

Table 3-17 List of Functions in “mtr_interrupt.c” (1/3)

File	Function	Process overview
mtr_interrupt.c	mtr_over_current_interrupt Input: none Output: none	Overcurrent detection process (detection from external circuit) - Disable INTP0 interruption - Call event process selection function (Error event occurs) - Error information setting (Overcurrent error (hardware))
	mtr_100usec_interrupt Input: none Output: none	Cycle timer interrupt (Call using INTTM01) Cycle: 100 µs - U, W phase current detection - Current detection offset correction process - Call overcurrent error monitoring process - Vector calculation - Call decoupling control process - Position/speed estimation calculation - Call current PI control process - Call deadtime compensation process - Call modulation process - Call PWM duty setting process - Call communication process
	mtr_1ms_interrupt Input: none Output: none	Cycle timer interrupt (Call using INTTM00) Cycle: 1 ms - Bus voltage detection - Calculation of inverse voltage - Startup control - Call command value setting process for d-axis and q-axis current and rotational speed - Call speed PI control process - Call error monitoring process
	mtr_calib_current_offset_uw (inline function) Input: st_mtr_tscs_t *st_tscs :: three-phase current detection structure pointer int16_t *s2_iuw_ad :: UVW current pointer Output: uint8_t :: current offset detection process completion flag	- Current offset detection process
	mtr_current_offset_adjustment_uvw (inline function) Input: st_mtr_tscs_t *st_tscs :: three-phase current detection structure pointer int16_t *s2_iuw_ad :: UVW current pointer int16_t s2_limit_over_current :: overcurrent limit value Output: uint16_t :: error status	Offset elimination process and overcurrent error detection
	mtr_transform_uvw_dq_abs (inline function) Input: int16_t *s2_uvw :: UVW phase pointer const st_mtr_ra_t *p_angle :: angular structure pointer int16_t *s2_dq :: dq axis pointer Output: none	UVW -> dq coordinate conversion
	mtr_pll_run (inline function) Input: st_mtr_pll_t *pmtr_pll :: PLL structure pointer int16_t s2_phase_error :: phase error [PU] Output: none	PLL loop process

Table 3-18 List of Functions in “mtr_interrupt.c” (2/3)

File	Function	Process overview
mtr_interrupt.c	mtr_bemf_magnitude (inline function) Input: int16_t s2_bemf_d :: d-axis induced voltage int16_t s2_bemf_q :: q-axis induced voltage Output: int16_t :: estimated value of induced voltage	Induced voltage calculation process Used for speed calculation when there is an open loop
	mtr_rotor_angle_update (inline function) Input: st_mtr_ra_t *p_angle :: angular structure pointer int16_t s2_angle_rad :: angle [PU] Output: none	Angle reflection process
	mtr_lpf1_run (inline function) Input: st_mtr_lpf1_t *st_lpf :: LPF structure pointer int16_t s2_input :: LPF input const uint8_t u1_q :: Q value of LPF Output: none	Primary LPF process
	mtr_current_pi_ctrl (inline function) Input: st_mtr_acr_t *st_acr :: ACR structure pointer Output: none	Current PI process
	mtr_decoupling (inline function) Input: st_mtr_acr_t *st_acr :: ACR structure pointer int16_t s2_speed_rad :: speed const st_mtr_parameter_t *p_mtr :: motor parameter structure pointer Output: none	Decoupling control process
	mtr_transform_dq_uvw_abs (inline function) Input: const int16_t *s2_dq :: dq axis pointer const st_mtr_ra_t *p_angle :: angular structure pointer int16_t *s2_uvw :: UVW phase pointer Output: none	dq -> UVW coordinate conversion
	mtr_deadtime_comp (inline function) Input: st_mtr_deadtime_comp_t *st_dtcomp :: deadtime compensation structure pointer st_mtr_mod_t *st_mod :: modulation structure pointer int16_t s2_vdc :: power supply voltage Output: none	Deadtime compensation process
	mtr_uvw_voltage_limit (inline function) Input: int16_t *s2_ref_v_uvw :: UVW phase voltage pointer int16_t s2_voltage_limit :: voltage limit value Output: none	Three-phase voltage limit processing
	mtr_mod (inline function) Input: st_mtr_mod_t *st_mod :: modulation structure pointer int16_t s2_reci_vdc :: inverse of voltage int16_t s2_voltage_limit :: voltage limit value Output: none	Modulation process

Table 3-19 List of Functions in “mtr_interrupt.c” (3/3)

File	Function	Process overview
mtr_interrupt.c	mtr_pwm_duty_ts (inline function) Input: st_mtr_tscs_t *st_tscs :: three-phase current detection structure pointer int16 s2_u :: u phase modulation factor int16 s2_v :: v phase modulation factor int16 s2_w :: w phase modulation factor Output: none	Duty calculation
	mtr_set_speed_ref (inline function) Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: int16 s2_speed_rad_ref_buff :: speed command value	Set command value for speed control
	mtr_pi_run (inline function) Input: st_mtr_pi_t *st_pi :: PI control structure pointer int16_t s2_err :: deviation const uint8_t u1_kp_q :: proportional gain shift value const uint8_t u1_kidt_q :: integral gain shift value Output: int16 s2_pi_out :: PI output	PI control process
	mtr_set_iq_ref (inline function) Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: int16 s2_iq_ref_buff :: q-axis current command value	Set q-axis current command value
	mtr_set_id_ref (inline function) Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: int16 s2_id_ref_buff :: d-axis current command value	Set d-axis current command value
	mtr_error_check Input: st_mtr_foc_t *st_foc :: FOC structure pointer Output: none	Error process - Overvoltage detection - Undervoltage detection - Excessive speed detection
	mtr_abs Input: int16_t s2_value :: input value Output: int16_t :: output value	Output absolute value of input
	mtr_limit_abs Input: int16_t s2_value :: input value int16_t s2_limit_value :: limit value Output: int16_t :: output value	Limit input by absolute value

3.3 List of Sensorless Vector Control Software Function Variables

A list of variables used in this control program is provided below. However, note that the local variables are not mentioned. Also, the control values in this control program are calculated after scaling each value. Regarding the variables to which the Q notation is applied, Qn in the scale field expresses that the factional part is n bits. However, the Q notation for some variables and structure members is calculated using definitions in r_mtr_scaling_parameter.h, so the default Q notation is written in the scale field in these cases. Variable/structure member units to which PU units are applied are written as [PU ([original unit]).

Table 3-20 List of Variables in “main.c”

Variable	Type	Qn	PU	Content	Remarks
g_u1_system_mode	static uint8_t	Q0	-	Mode system management	
g_u1_motor_status	static uint8_t	Q0	-	Motor status management	
g_u2_error_status	static uint16_t	Q0	-	Error status management	
g_u1_reset_req	static uint8_t	Q0	-	Reset request flag	0: No reset request 1: There is a reset request
g_u1_stop_req	static uint8_t	Q0	-	Stop request flag	
g_u1_flag_ui_change	static uint8_t	Q0	-	UI changing flag	0: GUI use (default) 1: Board user interface use
g_u1_flag_set_values	static uint8_t	Q0	-	Variable reflection flag	
g_u2_conf_hw	uint16_t	Q0	-	RMW configuration	
g_u2_conf_sw	uint16_t	Q0	-		
g_u2_conf_tool	uint16_t	Q0	-		
gui_u1_active_gui	uint8_t	Q0	-		
g_u2_conf_sw_ver	uint16_t	Q0	-		
com_s2_sw_userif	int16_t	Q0	-	UI management	0: ICS_UI 1: BOARD_UI
g_s2_sw_userif	int16_t	Q0	-		
com_u1_run_event	uint8_t	Q0	-	Change run mode	0: MTR_EVENT_STOP 1: MTR_EVENT_DRIVE 2: MTR_EVENT_ERROR 3: MTR_EVENT_RESET
g_u1_run_event	uint8_t	Q0	-		
g_u2_system_error	uint16_t	Q0	-	System error management	

Table 3-21 List of Variables in “r_mtr_board.c”

Variable	Type	Qn	PU	Content	Remarks
u1_sw_cnt	static uint8_t	Q0	-	Counter for judgment of chattering	

Table 3-22 List of Variables in “r_mtr_ics.c”

Variable	Type	Qn	PU	Content	Remarks
com_u1_direction	uint8_t	Q0	-	Direction of rotation	0: CW 1: CCW
com_f4_mtr_r	float	-	-	Resistance [Ω]	
com_f4_mtr_ld	float	-	-	d-axis inductance [H]	
com_f4_mtr_lq	float	-	-	q-axis inductance [H]	
com_f4_mtr_m	float	-	-	Induced voltage constant [Vs/rad]	
com_f4_mtr_j	float	-	-	Rotor inertia [kgm^2]	
com_u2_mtr_pp	uint16_t	Q0	-	Number of pole pairs	
com_u2_offset_calc_time	uint16_t	Q0	-	Current offset detection time	
com_s2_ref_speed_rpm	int16_t	Q0	-	Command rotational speed [rpm]	Mechanical angle
com_f4_ramp_limit_speed_rpm	float	-	-	Limit of acceleration [rpm/ms]	Mechanical angle
com_s2_max_speed_rpm	int16_t	Q0	-	Maximum speed [rpm]	Mechanical angle
com_f4_acr_nf_hz	float	Q0	-	Current PI control natural frequency [Hz]	
com_f4_asr_nf_hz	float	Q0	-	Speed PI control natural frequency [Hz]	
com_f4_asr_lpf_nf_hz	float	Q0	-	Speed LPF natural frequency [Hz]	
com_f4_pll_nf_hz	float	Q0	-	PLL natural frequency [Hz]	
com_s2_less2ol_speed_rpm	int16_t	Q0	-	Switching speed from sensorless to open loop [rpm]	Mechanical angle
com_s2_ol2less_speed_rpm	int16_t	Q0	-	Switching speed from open loop to sensorless [rpm]	Mechanical angle
com_f4_ol_ref_id	float	-	-	Open loop d-axis command current [A]	
com_f4_init_asr_intg	float	-	-	ASR integral term initial value during sensorless transition	
com_f4_ramp_limit_current	float	-	-	Limit value for current rise [A/ms]	
com_s2_enable_write	int16_t	Q0	-	Variable to allow variable rewriting	
g_s2_enable_write	int16_t	Q0	-	Variable to allow variable rewriting	
st_ics_input	mtr_ctrl_input_t	Q0	-	Structure for ICS variable transfer	Structure

Table 3-23 List of Variables in “r_mtr_driver_access.c”

Variable	Type	Qn	PU	Content	Remarks
st_ics_buff	mtr_ctrl_input_t	Q0	-	Buffer structure for ICS variable transfer	Structure
g_u1_trig_enable_write	uint8_t	Q0	-	Transfer completion flag	
g_u1_stop_req	uint8_t	Q0	-	Motor stop flag	
g_s2_cnt	int16_t	Q0	-	countor	

Table 3-24 List of Variables in “r_mtr_sc_table.c”

Variable	Type	Qn	PU	Content	Remarks
sc_table[1025]	static int16_t	Q0	-	Trinodal function table	

Table 3-25 List of Variables in “r_mtr_statemachine.c”

Variable	Type	Qn	PU	Content	Remarks
state_transition_table [MTR_SIZE_EVENT] [MTR_SIZE_STATE]	static uint8_t	Q0	-	Macro array for state transition	
action_table [MTR_SIZE_EVENT] [MTR_SIZE_STATE]	static mtr_action_t	Q0	-	Function array for state transition	

Table 3-26 List of Variables in “r_mtr_interrupt.c”

Variable	Type	Qn	PU	Content	Remarks
gst_foc	st_mtr_foc_t	Q0	-	Vector control structures	Structure
g_u1_cnt_ics	static uint8_t	Q0	-	Communication process cycle pixel skipping variable	

3.4 List of Sensorless Vector Control Software Structures

A list of structures used in this control program is provided below. Structures that are not used have been omitted.

Table 3-27 List of Variables in “r_mtr_parameter.h” / Structure: “st_mtr_parameter_t”

Variable	Type	Qn	PU	Content	Remarks
u2_mtr_pp	uint16_t	Q0	-	Number of pole pairs	
s2_mtr_r	int16_t	Q15	Resistance (voltage/current)	Resistance [PU]	
s2_mtr_ld	int16_t	Q15	Inductance (resistance/angular frequency)	d-axis inductance [PU]	
s2_mtr_lq	int16_t	Q15	Inductance (resistance/angular frequency)	q-axis inductance [PU]	
s2_mtr_m	int16_t	Q12	Induced voltage constant (voltage/angular frequency)	Induced voltage constant [PU]	
s2_mtr_j	int16_t	Q7	Inertia (Induced voltage constant × current × (number of pole pairs/angular frequency)^2)	Inertia [PU]	

Table 3-28 List of Variables in “r_mtr_parameter.h” / Structure: “st_mtr_design_parameter_t”

Variable	Type	Qn	PU	Content	Remarks
s2_acr_nf_hz	float	Q0	-	Current PI control natural frequency [Hz]	
s2_asr_nf_fz	float	Q0	-	Speed PI control natural frequency [Hz]	
s2_asr_lpf_nf_hz	float	Q0	-	Speed LPF natural frequency [Hz]	
s2_pll_nf_hz	float	Q0	-	PLL natural frequency [Hz]	
f4_dt	float	Q0	-	control period [sec]	
f4_dt_speed	float	Q0	-	control period for speed loop [sec]	
f4_r	float	Q0	-	Resistance [Ω]	
f4_ld	float	Q0	-	d-axis inductance [H]	
f4_lq	float	Q0	-	q-axis inductance [H]	
f4_m	float	Q0	-	Permanent magnetic flux [Wb]	
f4_j	float	Q0	-	Rotor inertia [[kgm^2]	
f4_pu_sf_afreq	float	Q0	-	frequency scale factor	
u1_q_pll_kp	uint8_t	Q0	-	Q-format of D-axis current PI proportional gain	
u1_q_pll_kidt	uint8_t	Q0	-	Q-format of D-axis current PI ki * dt	
u1_q_acr_kp	uint8_t	Q0	-	Q-format of Q-axis current PI proportional gain	
u1_q_acr_kidt	uint8_t	Q0	-	Q-format of Q-axis current PI ki * dt	
u1_q_asr_kp	uint8_t	Q0	-	Q-format of Speed current PI proportional gain	
u1_q_asr_kidt	uint8_t	Q0	-	Q-format of Speed current PI ki * dt	
u1_q_asr_lpf_k	uint8_t	Q0	-	Q-format of Speed LPF numerator	

Table 3-29 List of Variables in “r_mtr_parameter.h” / Structure: “st_mtr_ctrl_gain_t”

Variable	Type	Qn	PU	Content	Remarks
s2_acr_id_kp	int16_t	Q16	Resistance	d-axis current control proportional gain	
s2_acr_id_kidt	int16_t	Q16	Resistance	d-axis current control integral gain*operation period	
s2_acr_iq_kp	int16_t	Q16	Resistance	q-axis current control proportional gain	
s2_acr_iq_kidt	int16_t	Q16	Resistance	q-axis current control integral gain*operation period	
s2_asr_pi_kp	int16_t	Q14	Current/angular frequency	Speed control proportional gain	
s2_asr_pi_kidt	int16_t	Q18	Current/angular frequency	Speed control integral gain*operation period	
s2_asr_lpf_in_k	int16_t	Q14	-	Speed LPF input coefficient	
s2_pll_kp	int16_t	Q13	Angular frequency/angle	PLL proportional gain	
s2_pll_kidt	int16_t	Q17	1 angular frequency/angle	PLL integral gain*operation period	

Table 3-30 List of Variables in “r_mtr_driver_access.h” / Structure: “st_mtr_ctrl_input_t”

Variable	Type	Qn	PU	Content	Remarks
u1_direction	uint8_t	Q0	-	Direction of rotation	
u2_offset_calc_time	uint16_t	Q0	-	Offset detection time	
s2_ref_speed_rad	int16_t	Q14	Angular frequency	Reference rotational speed [PU]	Electric angle
s2_ramp_limit_speed_rad	int16_t	Q14	Angular frequency	Limit of acceleration [PU]	Electric angle
s2_max_speed_rad	int16_t	Q14	Angular frequency	Maximum speed [PU]	Electric angle
s2_less2ol_speed_rad	int16_t	Q14	Angular frequency	Switching speed from sensorless to open loop [PU]	Electric angle
s2_ol2less_speed_rad	int16_t	Q14	Angular frequency	Switching speed from open loop to sensorless [PU]	Electric angle
s2_ol_ref_id	int16_t	Q13	Current	Open loop d-axis command current [PU]	
s2_init_intg	int16_t	Q16	Current	ASR integral term initial value during sensorless transition	
s2_ramp_limit_current	int16_t	Q13	Current	Limit value for current rise [PU/ms]	
st_motor	st_mtr_parameter_t	-	-	Structure for motor parameter	Structure
st_gain	st_mtr_ctrl_gain_t	-	-	Structure for PI control	

Table 3-31 List of Variables in “r_mtr_statemachine.h” / Structure: “st_mtr_statemachine_t”

Variable	Type	Qn	PU	Content	Remarks
u1_status	uint8_t	Q0	-	Motor status	
u1_status_next	uint8_t	Q0	-	Next motor status	
u1_current_event	uint8_t	Q0	-	Execution event	

Table 3-32 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_lpf1_t”

Variable	Type	Qn	PU	Content	Remarks
s2_in_k	int16	Current: Q14 Speed: Q14	-	LPF input gain	
s2_out_k	int16	Current: Q14 Speed: Q14	-	LPF previous gain	
s2_pre_out	int16	Current: Q13 Speed: Q14	Current: current Speed: angular frequency	Previous output value	

Table 3-33 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_pi_t”

Variable	Type	Qn	PU	Content	Remarks
s2_kp	int16_t	Current: Q16 Speed: Q14 PLL: Q13	-	Proportional gain	
s2_kidt	int16_t	Current: Q16 Speed: Q18 PLL: Q17	-	Integral gain x control period	
s2_intg	int16_t	Current: Q13 Speed: Q14 PLL: Q14	Current: Resistance Speed: Current/angular frequency PLL: Angular frequency/angle	Integral term	
s2_ilimit	int16_t	Current: Q13 Speed: Q14 PLL: Q14	Current: Resistance Speed: Current/angular frequency PLL: Angular frequency/angle	Integral limit (up/down symmetry)	

Table 3-34 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_acr_t”

Variable	Type	Qn	PU	Content	Remarks
s2_ctrl_period	int16_t	Q18	Time	Current control cycle	
s2_ref_vd	int16_t	Q13	Voltage	d-axis output voltage command value	
s2_ref_vq	int16_t	Q13	Voltage	q-axis output voltage command value	
s2_pre_ref_vd	int16_t	Q13	Voltage	Previous d-axis output voltage command value	
s2_pre_ref_vq	int16_t	Q13	Voltage	Previous q-axis output voltage command value	
s2_id_ad	int16_t	Q13	Current	d-axis current	
s2_iq_ad	int16_t	Q13	Current	q-axis current	
s2_ref_id	int16_t	Q13	Current	d-axis current command	
s2_ref_iq	int16_t	Q13	Current	q-axis current command	
s2_ref_id_ctrl	int16_t	Q13	Current	d-axis current command control value	
s2_ref_iq_ctrl	int16_t	Q13	Current	q-axis current command control value	
s2_limit_iq	int16_t	Q13	Current	q-axis current limit	
s2_ol_ref_id	int16_t	Q13	Current	Open loop d-axis current command value	
s2_ramp_limit_current	int16_t	Q13	Current	Limit value for current rise [PU/ms]	
s2_iq_lpf	int16_t	Q13	Current	q-axis current LPF value	
st_iq_lpf	st_mtr_lpf1_t	-	-	q-axis current LPF structure	Structure
st_pi_id	st_mtr_pi_t	-	-	d-axis current PI structure	
st_pi_iq	st_mtr_pi_t	-	-	q-axis current PI structure	

Table 3-35 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_pll_t”

Variable	Type	Qn	PU	Content	Remarks
s2_dt	int16_t	Q18	Time	Control cycle	
s2_speed_rad	int16_t	Q14	Frequencies	Speed	
st_pi	st_mtr_pi_t		-	PI structure	Structure

Table 3-36 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_ra_t”

Variable	Type	Qn	PU	Content	Remarks
s2_rotor_angle_rad;	int16_t	Q12	Angle	Angle	
s2_rotor_angle_rad_ctrl	int16_t	Q12	Angle	Angle control value	
s2_sin;	int16_t	Q12	-	Sin	
s2_cos;	int16_t	Q12	-	Cos	

Table 3-37 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_deadtime_comp_t”

Variable	Type	Qn	PU	Content	Remarks
s2_deadtime_error_voltage	int16_t	Q12	Voltage	Voltage error	
s2_deadtime_limit_current	int16_t	Q12	Current	Current limit	
s2_ref_i_uvw[3]	int16_t	Q12	Current	Three-phase command current	
s2_delta_v_uvw[3]	int16_t	Q12	Voltage	Three-phase voltage compensation value	

Table 3-38 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_deadtime_comp_t”

Variable	Type	Qn	PU	Content	Remarks
s2_deadtime_error_voltage	int16_t	Q12	Voltage	Voltage error	
s2_deadtime_limit_current	int16_t	Q12	Current	Current limit	
s2_ref_i_uvw[3]	int16_t	Q12	Current	Three-phase command current	
s2_delta_v_uvw[3]	int16_t	Q12	Voltage	Three-phase voltage compensation value	

Table 3-39 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_deadtime_comp_t”

Variable	Type	Qn	PU	Content	Remarks
u1_ref_dir;	int16_t	-	-	Direction of rotation command	0: CW 1: CCW
s2_speed_ctrl_period	int16_t	Q15	Time	Speed control cycle	
s2_ref_speed_rad;	int16_t	Q14	Angular frequency	Command rotational speed	
s2_ref_speed_rad_ctrl;	int16_t	Q14	Angular frequency	Command speed control value	
s2_speed_rad_origin;	int16_t	Q14	Angular frequency	Speed (no filter)	
s2_speed_rad;	int16_t	Q14	Angular frequency	Speed (with filter)	
s2_ramp_limit_speed_rad	int16_t	Q14	Angular frequency	Limit of acceleration	
s2_max_speed_rad;	int16_t	Q14	Angular frequency	Maximum speed	
s2_limit_speed_rad;	int16_t	Q14	Angular frequency	Limit of speed	
s2_init_intg;	int16_t	Q13	Current	Integral term initial value during sensorless switching	
s2_less2ol_speed_rad;	int16_t	Q14	Angular frequency	Switching speed from sensorless to open loop	
s2_ol2less_speed_rad;	int16_t	Q14	Angular frequency	Switching speed from open loop to sensorless	
st_pi;	st_mtr_pi_t	-	-	Speed PI structure	Structure
st_lpf;	st_mtr_lpf1_t	-	-	Speed LPF structure	

Table 3-40 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_mod_t”

Variable	Type	Qn	PU	Content	Remarks
s2_ref_vu	int16_t	Q13	Voltage	U phase voltage command	
s2_ref_vv	int16_t	Q13	Voltage	V phase voltage command	
s2_ref_vw	int16_t	Q13	Voltage	W phase voltage command	
s2_com_v	int16_t	Q13	Voltage	Voltage offset	
s2_mod_u;	int16_t	Q12	-	U phase modulation factor	
s2_mod_v;	int16_t	Q12	-	V phase modulation factor	
s2_mod_w;	int16_t	Q12	-	W phase modulation factor	

Table 3-41 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_tscs_t”

Variable	Type	Qn	PU	Content	Remarks
s2_duty_u	int16_t	Q0	-	U phase duty (PWM register setting)	
s2_duty_v	int16_t	Q0	-	V phase duty (PWM register setting)	
s2_duty_w	int16_t	Q0	-	W phase duty (PWM register setting)	
s2_offset_iu	int16_t	Q13	Current	U phase current offset value	
s2_offset_iw	int16_t	Q13	Current	W phase current offset value	
s4_offset_iu_sum	int32_t	Q13	Current	U phase current offset value integral value	
s4_offset_iw_sum	int32_t	Q13	Current	W phase current offset value integral value	
u2_offset_calc_time	uint16_t	-	-	Offset current measurement count	
u2_offset_sample_cnt	uint16_t	-	-	Offset current measurement sample count	
u2_crnt_ad[2]	uint16_t	-	-	UW phase current A/D conversion value	

Table 3-42 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_est_phe_t”

Variable	Type	Qn	PU	Content	Remarks
s2_ed	int16_t	Q13	Voltage	d-axis induced voltage	
s2_eq	int16_t	Q13	Voltage	q-axis induced voltage	
s2_e	int16_t	Q13	Voltage	Induced voltage	
s2_phase_err_rad	int16_t	Q12	Angle	Phase error	
s2_r_id	int16_t	Q13	Voltage	R*id	
s2_r_iq	int16_t	Q13	Voltage	R*iq	
s2_speed_ld_id	int16_t	Q13	Voltage	Speed*Ld*id	
s2_speed_lq_iq	uint16_t	Q13	Voltage	Speed*Lq*iq	

Table 3-43 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_foc_t” (1/2)

Variable	Type	Qn	PU	Content	Remarks
u2_run_mode	uint16_t	-	-	Operating modes	0x00: Init mode 0x01: Boot mode 0x02: Drive mode 0x03: Analysis mode 0x04: Tune mode
u2_ctrl_conf	uint16_t	-	-	Control inputs	0x01: Current control 0x02: Speed control 0x04: Position control 0x08: Torque control 0x10: Voltage control
u2_error_status	uint16_t	-	-	Error status	0x0000: No error 0x0001: Overcurrent error (hardware) 0x0002: Overvoltage error 0x0004: Rotational speed error 0x0008: Hall timeout error 0x0010: Induced voltage timeout error 0x0020: Hall pattern error 0x0040: Induced voltage pattern error 0x0080: Undervoltage error 0x0100001: Overcurrent error (software) 0xFFFF: Undefined error
u1_direction	uint8_t	-	-	Current direction of rotation	0: CW 1: CCW
u1_flag_charge_cap	uint8_t	-	-	Current offset value calculation flag	0: Execute offset calculation process 1: Offset calculation process completed
u1_state_offset_calc	uint8_t	-	-	Offset detection state fallen flag	0: Eliminating offset 1: Offset elimination process completed 2: Offset elimination completed
u1_state_ref_id	uint8_t	-	-	d-axis current command value generation status	0: d-axis current increase 1: d-axis current constant 2: d-axis current decrease 3: d-axis current 0
u1_state_ref_iq	uint8_t	-	-	q-axis current command value generation status	0: q-axis current 0 1: Speed PI output 2: q-axis current decrease
u1_state_ref_speed	uint8_t	-	-	Speed command value generation status	0: Speed 0 1: Speed change
u1_flag_down_to_ol	uint8_t	-	-	Open loop transition flags	0: No transition 1: Execute transition
s2_iu_ad	int16_t	Q13	Current	U phase current	
s2_iv_ad	int16_t	Q13	Current	V phase current	
s2_iw_ad	int16_t	Q13	Current	W phase current	
s2_vdc_ad	int16_t	Q13	Voltage	Power source voltage	
s2_reci_vdc	int16_t	Q13	1/voltage	Inverse of voltage	
s2_limit_vout	int16_t	Q13	Voltage	Voltage limit	
s2_limit_over_current	int16_t	Q13	Current	Overcurrent limit value	
s2_limit_over_voltage	int16_t	Q13	Voltage	Overvoltage limit value	
s2_limit_under_voltage	int16_t	Q13	Voltage	Undervoltage limit value	

Table 3-44 List of Variables in “r_mtr_foc_less_speed.h” / Structure: “st_mtr_foc_t” (2/2)

Variable	Type	Qn	PU	Content	Remarks
st_stm	st_mtr_statemachine_t	-	-	Structure for state machine	
st_motor	st_mtr_parameter_t	-	-	Structure for motor parameter	
st_ra	st_mtr_ra_t	-	-	Structure for rotor angle	
st_phe	st_mtr_est_phe_t	-	-	Structure for phase error estimate	
st_tscs	st_mtr_tscs_t	-	-	Structure for three-phase current detection	
st_acr	st_mtr_acr_t	-	-	ACR structure	Current PI control
st_asr	st_mtr_asr_t	-	-	ASR structure	Speed PI control
st_mod	st_mtr_mod_t	-	-	Structure for modulation	
st_pll	st_mtr_pll_t	-	-	Structure for PLL control	
st_gain_buf	st_mtr_ctrl_gain_t	-	-	Structure for control gain	
st_dt_comp	st_mtr_deadtime_comp_t	-	-	Structure for deadtime compensation	

3.5 List of Sensorless Vector Control Software Macro Definitions

A list of macro definitions used in this control program is provided below.

Table 3-45 List of Macro Definitions in “r_mtr_config.h”

Macro	Definition value	Description	Remarks
IP_MRSSK	-	Select inverter board	
MP_TG55L	-	Select motor parameters	
CP_TG55L	-	Select control parameters	
ICS_UI	0	RMW UI	Default
BOARD_UI	1	RSSK board UI	
MTRCONF_DEFAULT_UI	0:1	Select UI	Board UI / ICS UI
USE_DEADTIME_COMP	0:1	Select deadtime compensation process	Default setting 0
USE_SPEED_LPF	0:1	Select speed LPF	Default setting 1
USE_CURRENT_LPF	0:1	Select current PPF	Default setting 1
MOD_3PH_SPWM	0	Sine wave modulation	
MOD_3PH_TOW	1	Third harmonic calculation	
MOD_METHOD	0:1	Modulation method	Default setting 1
SC_LIB	0	Trinodal function library	
SC_TABLE	1	Trinodal function table	
SC_METHOD	0:1	Trinodal function operation system	Default setting 0

Table 3-46 List of Macro Definitions in “motor_parameter.h”

Macro	Definition value	Description	Remarks
MP_POLE_PAIRS	2	Number of pole pairs	
MP_RESISTANCE	9.125f	Resistance [Ω]	
MP_D_INDUCTANCE	0.003844f	d-axis inductance [H]	
MP_Q_INDUCTANCE	0.004315f	q-axis inductance [H]	
MP_MAGNETIC_FLUX	0.02144f	Magnetic flux [Wb]	
MP_ROTOR_INERTIA	0.000002050f	Inertia [kgm^2]	
MP_NOMINAL_CURRENT_RMS	0.42f	Nominal current [A]	
MP_RATED_SPEED	2650	Rated speed [rpm]	

Table 3-47 List of Macro Definitions in “control_parameter.h”

Macro	Definition value	Description	Remarks
CP_ACR_NF_HZ	300.0f	Current PI control natural frequency [Hz]	
CP_ASR_NF_HZ	30.0f	Speed PI control natural frequency [Hz]	
CP_PLL_NF_HZ	MP_RATED_SPEED	PLL control natural frequency [Hz]	
CP_MAX_SPEED_RPM	MP_RATED_SPEED*1.5	Maximum speed (mechanical angle) [rpm]	
CP_SPEED_LIMIT_RPM	300.0f	Limit of speed (mechanical angle) [rpm]	
CP_OC_LIMIT	1.5		
CP_OL_REF_ID	MP_NOMINAL_CURRENT_RMS	d-axis current command value [A]	
CP_INIT_ASR_INTEG	0.2f	q-axis current PI integral term PI initial value [A]	
CP_LAMP_LIMIT_CURRENT	0.01f	Limit value for current rise [PU/ms]	
CP_OL2LESS_SPEED_RPM	MP_RATED_SPEED*0.4	Switching speed from sensorless to open loop (mechanical angle) [rpm]	
CP_LESS2OL_SPEED_RPM	MP_RATED_SPEED*0.3	Switching speed from open loop to sensorless (mechanical angle) [rpm]	
CP_LAMP_LIMIT_SPEED_RPM	1	Limit of acceleration [rpm/ms]	
CP_OFFSET_CALC_TIME	384.0f	Current offset value calculation time [ms]	

Table 3-48 List of Macro Definitions in “r_mtr_inverter_parameter.h”

Macro	Definition value	Description	Remarks
IP_DEADTIME	2.0f	Deadtime	
IP_CURRENT_RANGE	20.0f	Current scaling range [A]	
IP_VDC_RANGE	111.0f	Voltage scaling range [V]	
IP_INPUT_V	24.0f	Input voltage [V]	
IP_CURRENT_LIMIT	2.0f	Current limit value [A]	
IP_OVERVOLTAGE_LIMIT	28.0f	Overvoltage limit [V]	
IP_UNDERVOLTAGE_LIMIT	15.0f	Undervoltage limit [V]	

Table 3-49 List of Macro Definitions “r_mtr_scaling_parameter.h”

Macro	Definition value	Description	Remarks
FP_SF_VOLTAGE	37	Voltage PU conversion value ((((IP_VDC_RANGE/1023)*PU_SF_VOLTAGE) * (1<<MTR_Q_VOLTAGE))	
FP_SF_CURRENT	381	Current PU conversion value ((((IP_CURRENT_RANGE/1023)*PU_SF_CURRENT) * (1<<MTR_Q_CURRENT))	
PU_BASE_CURRENT_A	MP_NOMINAL_CURRENT_RMS	Current standard value [A]	
PU_BASE_VOLTAGE_V	IP_INPUT_V	Voltage standard value [A]	
PU_BASE_ANGLE_Rad	1.0f	Angle standard value [rad]	
PU_BASE_FREQ_Hz	MTR_TWOPI*CP_MAX_SPEED_RPM *MP_POLE_PAIRS/60	Frequency standard value [Hz]	
PU_SF_CURRENT	1.0f / PU_BASE_CURRENT_A	Current scale [PU/A]	
PU_SF_VOLTAGE	1.0f / PU_BASE_VOLTAGE_V	Voltage scale [PU/V]	
PU_SF_AFREQ	1.0f / PU_BASE_FREQ_Hz	Angular frequency scale [PU/(rad/s)]	
PU_SF_ANGLE	1.0f / PU_BASE_ANGLE_Rad	Angle scale [PU/rad]	
PU_SF_TIME	PU_SF_ANGLE / PU_SF_AFREQ	Time scale [PU/s]	
PU_SF_RES	PU_SF_VOLTAGE / PU_SF_CURRENT	Resistance scale [PU/ohm]	
PU_SF_IND	PU_SF_RES / PU_SF_AFREQ	Inductance scale [PU/H]	
PU_SF_FLUX	PU_SF_VOLTAGE / PU_SF_AFREQ	Magnetic flux scale [PU/Wb]	
PU_SF_INERTIA	PU_SF_FLUX * PU_SF_CURRENT / (MP_POLE_PAIRS * MP_POLE_PAIRS * PU_SF_AFREQ * PU_SF_AFREQ)	Inertia scale [PU/(rad/kgm^2)]	
PU_SF_RPM_RAD	1.0f / CP_MAX_SPEED_RPM	Scale of conversion from [rpm] to [rad/s]	
PU_SF_RAD_RPM	CP_MAX_SPEED_RPM	Scale of conversion from [rad/s] to [rpm]	
PU_SF_ACR_KP	PU_SF_RES	Current PI proportional gain scale	
PU_SF_ACR_KIDT	PU_SF_RES	Current PI integral gain scale	
PU_SF_ASR_KP	PU_SF_CURRENT / PU_SF_AFREQ	Speed PI proportional gain scale	
PU_SF_ASR_KIDT	PU_SF_CURRENT / PU_SF_AFREQ	Speed PI integral gain scale	
PU_SF_PLL_KP	PU_SF_AFREQ / PU_SF_ANGLE	PLL proportional gain scale	
PU_SF_PLL_KIDT	PU_SF_AFREQ / PU_SF_ANGLE	PLL integral gain scale	
MTR_Q_ANGLE	12	Q-format of angle	
MTR_Q_CTRL_TIME	17	Q-format of FOC control cycle	
MTR_Q_CTRL_TIME_SPEED	14	Q-format of speed control cycle	
MTR_Q_CURRENT	13	Q-format of current	
MTR_Q_VOLTAGE	13	Q-format of voltage	
MTR_Q_AFREQ	14	Q-format of angular frequency	
MTR_Q_RESISTANCE	17	Q-format of resistance	
MTR_Q_INDUCTANCE	17	Q-format of inductance	
MTR_Q_FLUX	15	Q-format of magnetic flux	
MTR_Q_INERTIA	13	Q-format of inertia	
MTR_Q_ZETA	9	Q-format of attenuation coefficient	
MTR_Q_VMOD	12	Q-format of PWM modulation factor	
MTR_Q_RECIV	13	Q-format of inverse voltage	
MTR_Q_FLUX_INV	16	Q-format of inertia	
MTR_Q_DIV_DSP	12	Q-format of DSP function division	
MTR_Q_SIN_COS_DSP	12	Q-format of trinodal function of DSP function	

Table 3-50 List of Macro Definitions “r_mtr_scaling_parameter.h”

Macro	Definition value	Description	Remarks
MTR_Q_ACR_KP	18	Q-format of speed PI proportional gain	
MTR_Q_ACR_KIDT	14	Q-format of speed PI integral gain * control period	
MTR_Q_ASR_KP	12	Q-format of current PI proportional gain	
MTR_Q_ASR_KIDT	15	Q-format of current PI integral gain * control period	
MTR_Q_SPEED_LPF_CO	14	Q-format of PLL proportional gain	
MTR_Q_ACR_KP	18	Q-format of PLL integral gain * control period	

Table 3-51 List of Macro Definitions in “main.h”

Macro	Definition value	Description	Remarks
MODE_INACTIVE	0x00	Inactive mode	
MODE_ACTIVE	0x01	Active mode	
MODE_ERROR	0x02	Error mode	
SIZE_STATE	3	Number of modes	

Table 3-52 List of Macro Definitions in “ICS_define.h”

Macro	Definition value	Description	Remarks
RL78	-	CPU definition	

Table 3-53 List of Macro Definitions in “r_mtr_ics.h”

Macro	Definition value	Description	Remarks
MTR_ICS_DECIMATION	2	Number of pixels skipped in ICS processing	
ICS_ADDR	0xFE00	Address of ICS	
ICS_INT_LEVEL	2	ICS interrupt level setting	
ICS_NUM	0x40	Data size of ICS communication	
ICS_BRR	15	ICS bit rate register selection	
ICS_MODE	0	ICS interrupt mode setting	

Table 3-54 List of Macro Definitions in “r_mtr_board.h”

Macro	Definition value	Description	Remarks
SW_CHATTERING_CNT	10	Count for judgement to remove chattering	
VR1_MARGIN	400	Margin value for VR1	
VR1_SCALING	(CP_MAX_SPEED_RPM + VR1_MARGIN) / 0x0200	VR1 speed command value calculation constant	
VR1_OFFSET	0x1FF	VR1 offset adjustment constant	
VR1_MIN_SPEED	100	inimum speed of VR1 [rpm]	

Table 3-55 List of Macro Definitions in “r_mtr_ctrl_rl78g1f.h” [1/2]

Macro	Definition value	Description	Remarks
MTR_INT_DECIMATION	1	Interrupt processing carrier pixel skipping	
MTR_PWM_TIMER_FREQ	64.0f	PWM timer frequency [kHz]	
MTR_INTVAL_TIMER_FREQ	32.0f	Interval timer frequency [kHz]	
MTR_CARRIER_FREQ	20.0f	Carrier interrupt frequency [kHz]	
MTR_INVTVAL_PERIOD	(MTR_INT_DECIMATION + 1) * 1000.0f / (MTR_CARRIER_FREQ)	Interval timer cycle [µs]	
MTR_DEADTIME	IP_DEADTIME	Deadtime [µs]	
MTR_DEADTIME_CNT	(int16)(MTR_DEADTIME * MTR_PWM_TIMER_FREQ)	Deadtime settings	
MTR_CARRIER_CNT	(uint16_t)(MTR_PWM_TIMER_FREQ * 1000 / MTR_CARRIER_FREQ * 0.5f)	Carrier settings	
MTR_HALF_CARRIER_CNT	(uint16)(MTR_CARRIER_SET * 0.5f)	Carrier settings (intermediate value)	
MTR_CURRENT_ADCONV_TIME	6.0f	Time [µs] taken for two-phase current A/D conversion	
MTR_AD_TIME_ADJUST	2.55f	AD timing adjustment	
MTR_AD_TIME_CNT	(uint16)(MTR_PWM_TIMER_FREQ * MTR_PWM_LA_MIN_ONTIME)	A/D conversion time counter value g	
MTR_CENTER_AMPLITUDE_CNT	(uint16_t)((MTR_CARRIER_CNT-(MTR_AD_TIME_CNT+MTR_DEADTIME_CNT)) * 0.5f + MTR_AD_TIME_CNT+MTR_DEADTIME_CNT)	PWM timer center amplitude	
MTR_VOLTAGE_LIMIT_OFFSET	(int16_t)(((MTR_CURRENT_ADCONV_TIME + MTR_DEADTIME * 2) / (1000/MTR_CARRIER_FREQ)) * 0.5f * (1 << MTR_Q_VOLTAGE))	Voltage offset limit [PU (V)]	
MTR_VOLTAGE_ERROR	(MTR_DEADTIME/1000000) * (MTR_CARRIER_FREQ * 1000)/2	Deadtime compensation coefficient	
MTR_DEADTIME_CURRENT_LIMIT	MP_NOMINAL_CURRENT_RMS * 0.1f	Current limit value	
MTR_CTRL_PERIOD	(MTR_INT_DECIMATION + 1) / (MTR_CARRIER_FREQ * 1000)	Current control cycle	
MTR_SPEED_CTRL_PERIOD	0.001f	Speed control cycle	
MTR_PORT_UP	P1_bit.no5	U phase (positive phase) voltage output port	
MTR_PORT_UN	P1_bit.no4	U phase (negative phase) voltage output port	
MTR_PORT_VP	P1_bit.no3	V phase (positive phase) voltage output port	
MTR_PORT_VN	P1_bit.no1	V phase (negative phase) voltage output port	
MTR_PORT_WP	P1_bit.no2	W phase (positive phase) voltage output port	
MTR_PORT_WN	P1_bit.no0	W phase (negative phase) voltage output port	
MTR_PORT_ENC_A	P0_bit.no0	Encoder A phase input port	
MTR_PORT_ENC_B	P0_bit.no1	Encoder B phase input port	
MTR_PORT_ENC_Z	P5_bit.no0	Encoder Z phase input port	
MTR_PORT_HALL_U	P1_bit.no2	U phase Hall effect sensor input port	
MTR_PORT_HALL_V	P1_bit.no3	V phase Hall effect sensor input port	
MTR_PORT_HALL_W	P1_bit.no4	W phase Hall effect sensor input port	
MTR_PORT_SW1	P12_bit.no5	SW1 input port	
MTR_PORT_SW2	P13_bit.no7	SW2 input port	
MTR_PORT_LED1	P14_bit.no1	LED1 output port	
MTR_PORT_LED2	P14_bit.no0	LED2 output port	
MTR_PORT_LED3	P0_bit.no4	LED3 output port	

Table 3-56 List of Macro Definitions in “r_mtr_ctrl_rl78g1f.h” [2/2]

Macro	Definition value	Description	Remarks
MTR_DUTY_U	TRDGRD0	Timer RD general register	
MTR_DUTY_V	TRDGRC1	Timer RD general register	
MTR_DUTY_W	TRDGRD1	Timer RD general register	
MTR_ADCCH_VR1	6	A/D converter channel of VR1	
MTR_ADCCH_VDC	4	A/D converter channel of bus voltage	
MTR_ADCCH_VU	16	A/D converter channel of U phase voltage	
MTR_ADCCH_VV	0	A/D converter channel of V phase voltage	
MTR_ADCCH_VW	1	A/D converter channel of W phase voltage	
MTR_ADCCH_IU	2	A/D converter channel of U phase current	
MTR_ADCCH_IV	19	A/D converter channel of V phase current	
MTR_ADCCH_IW	3	A/D converter channel of W phase current	
MTR_ADC_DATA_SHIFT	6	A/D conversion value shift amount	
MTR_ADC_OFFSET	0x1FF	A/D conversion value offset	
ERROR_NONE	0x00	No error	
ERROR_CHANGE_CLK_TIMEOUT	0x01	Timeout error for clock settings	
ERROR_CHARGE_CAP_TIMEOUT	0x02	Capacitor charging timeout error	

Table 3-57 List of Macro Definitions in “r_mtr_common.h”

Macro	Definition value	Description	Remarks
MTR_TWOPI	2*3.14159265359f	2π	
MTR_SQRT_3	1.7320508f	√3	
MTR_CW	0	CW	
MTR_CCW	1	CCW	
MTR_ON	0	ON	
MTR_OFF	1	OFF	
MTR_CLR	0	Flag clear	
MTR_SET	1	Flag set	

Table 3-58 List of Macro Definitions in “r_mtr_parameter.h”

Macro	Definition value	Description	Remarks
MTR_PWM_DUTY_RANGE	4095	Duty range	
MTR_INPUT_V	IP_INPUT_V	Input voltage	
MTR_HALF_VDC	MTR_INPUT_V * 0.5f	50% of voltage	
MTR_MCU_ON_V	MTR_INPUT_V * 0.8f	80% of voltage	
MTR_VDC_SCALING	IP_VDC_RANGE / 1023.0f	Voltage scaling value	
MTR_OVERVOLTAGE_LIMIT	IP_OVERVOLTAGE_LIMIT	Overvoltage limit value	
MTR_UNDERVOLTAGE_LIMIT	IP_UNDERVOLTAGE_LIMIT	Undervoltage limit value	
MTR_ANGLE_RANGE	(int16_t)(MTR_TWOPI* PU_SF_ANGLE * 4096)	Angle range 2π	
MTR_ANGLE_HALF_RANGE	(int16_t)(MTR_ANGLE_RANGE/2))	Angle range π	
MTR_ANGLE_QUAT_RANGE	(int16_t)(MTR_ANGLE_RANGE/4))	Angle range π/2	
MTR_CURRENT_SCALING	IP_CURRENT_RANGE / 1023.0f	Current scaling value	
MTR_OVERCURRENT_LIMIT	IP_CURRENT_LIMIT	Current limit value	
MTR_I_LIMIT_VD	IP_INPUT_V * 0.5f	Vd current PI limit	
MTR_I_LIMIT_VQ	IP_INPUT_V * 0.5f	Vq current PI limit	
MTR_RPM_RAD	(MP_POLE_PAIRS * MTR_TWOPI) / 60.0f	Conversion from [rpm] to [rad/s]	
MTR_SPEED_LIMIT_RAD	CP_SPEED_LIMIT_RPM * MTR_RPM_RAD	Speed limit value [rad/s]	
MTR_MAX_SPEED_RAD	CP_MAX_SPEED_RPM * MTR_RPM_RAD	Maximum speed [rad/s]	
MTR_LIMIT_IQ	MP_NOMINAL_CURRENT_RMS * MTR_SQRT_3	Speed PI output limit value	
MTR_I_LIMIT_IQ	MP_NOMINAL_CURRENT_RMS * MTR_SQRT_3	Limit value for speed PI integral term output	
MTR_LESS2OL_SPEED_RAD	CP_LESS2OL_SPEED_RPM * MTR_RPM_RAD	Switching speed from sensorless to open loop [rad/s]	
MTR_OL2LESS_SPEED_RAD	CP_OL2LESS_SPEED_RPM * MTR_RPM_RAD	Switching speed from open loop to sensorless [rad/s]	
MTR_FLUX_INV	(1.0f/(MP_MAGNETIC_FLUX * PU_SF_FLUX))	1/magnetic flux [PU]	

Table 3-59 List of Macro Definitions in“r_mtr_sc_table.h”

Macro	Definition value	Description	Remarks
MTR_PI_2	1024	π / 2	
MTR_PI	2048	π	
MTR_3PI_2	3072	3π/2	

Table 3-60 List of Macro Definitions in “r_mtr_statemachine.h”

Macro	Definition value	Description	Remarks
MTR_MODE_INIT	0x00	Initialization mode	
MTR_MODE_DRIVE	0x01	Drive mode	
MTR_MODE_STOP	0x02	Stop mode	
MTR_SIZE_STATE	3	Number of states	
MTR_EVENT_STOP	0x00	Stop event	
MTR_EVENT_DRIVE	0x01	Run event	
MTR_EVENT_ERROR	0x02	Error event	
MTR_EVENT_RESET	0x03	Reset event	
MTR_SIZE_EVENT	4	Number of events	

Table 3-61 List of Macro Definitions in “r_mtr_foc_less_speed.h”

Macro	Definition value	Description	Remarks
MTR_CONTROL_CURRENT	0x01	Current control	
MTR_CONTROL_SPEED	0x02	Speed control	
MTR_CONTROL_POSITION	0x04	Position control	
MTR_CONTROL_TORQUE	0x08	Torque control	
MTR_CONTROL_VOLTAGE	0x10	Voltage control	
MTR_ERROR_NONE	0x00	No error	
MTR_ERROR_OVER_CURRENT	0x01	Overcurrent error	
MTR_ERROR_OVER_VOLTAGE	0x02	Overvoltage error	
MTR_ERROR_OVER_SPEED	0x04	Excessive speed error	
MTR_ERROR_HALL_TIMEOUT	0x08	Hall timeout error	
MTR_ERROR_BEMF_TIMEOUT	0x10	Induced voltage timeout error	
MTR_ERROR_HALL_PATTERN	0x20	Hall pattern error	
MTR_ERROR_BEMF_PATTERN	0x40	Induced voltage pattern error	
MTR_ERROR_UNDER_VOLTAGE	0x80	Undervoltage error	
MTR_ERROR_UNKNOWN	0xff	Undefined error	
MTR_ID_ZERO_CONST	0	d-axis current 0 control	
MTR_ID_MANUAL	1	d-axis current manual control	
MTR_IQ_ZERO_CONST	0	q-axis current 0 control	
MTR_IQ_MANUAL	1	q-axis current manual control	
MTR_IQ_SPEED_PI_OUTPUT	2	Speed PI control output	
MTR_SPEED_ZERO_CONST	0	Speed 0 control	
MTR_SPEED_MANUAL	1	Speed manual control	
MTR_OFFSET_CALC_EXE	0	Execute offset elimination	
MTE_OFFSET_CALC_END	1	Terminate offset elimination	
MTR_OFFSET_CALC_DONE	2	Complete offset elimination process	

Table 3-62 List of Macro definitions in “r_mtr_est_phase_err.h”

Macro	Definition value	Description	Remarks
MTR_TWOPI_Q	(int16_t)(4096.0f * MTR_TWOPI)	2π (Q-format of angle)	
MTR_INV_TWOPI_Q	(int16_t)(4096.0f / MTR_TWOPI)	1/ 2π (Q-format of angle)	

3.6 Control flows (flowcharts)

3.6.1 Main process

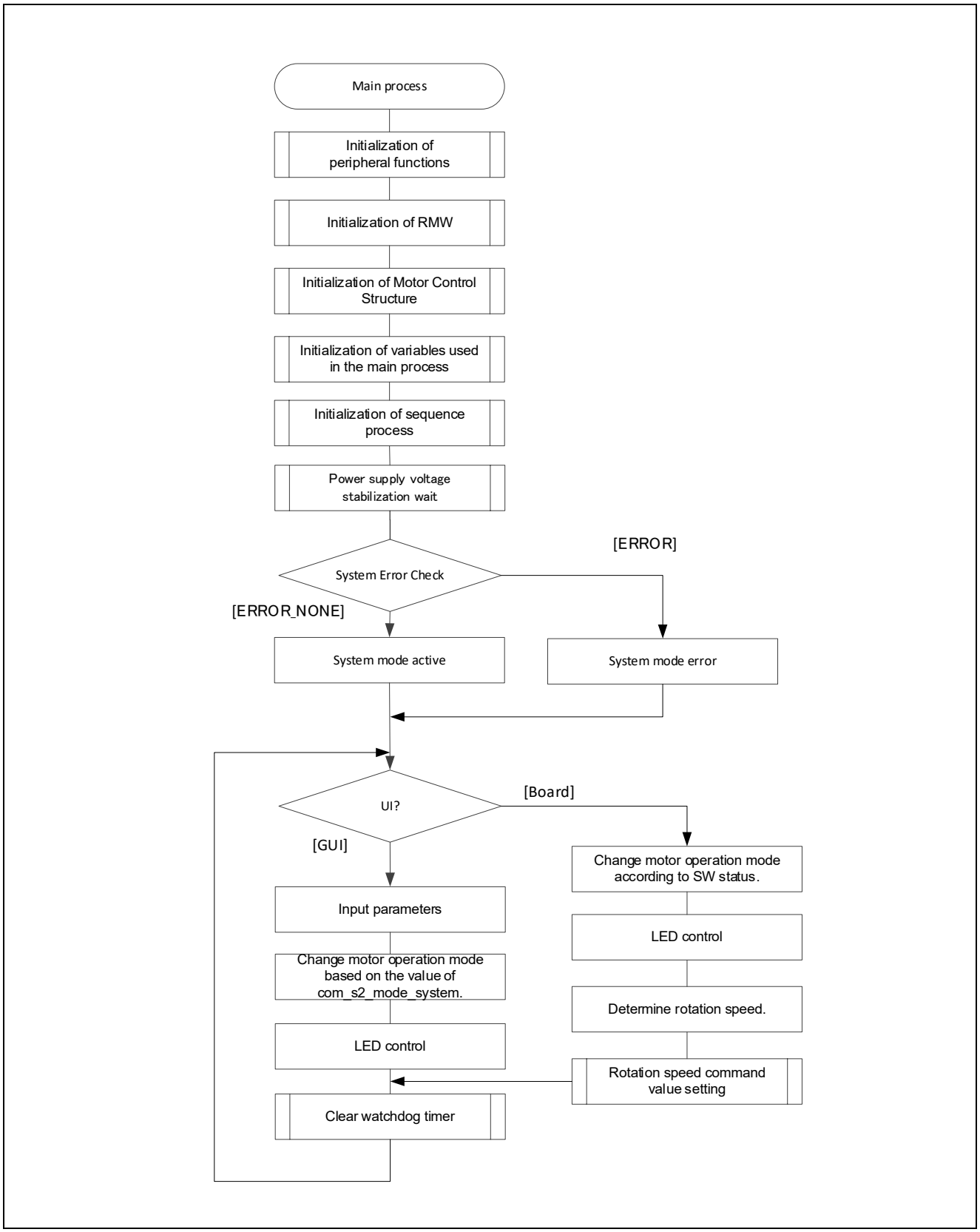

Figure 3-7 Main Process Flowchart

3.6.2 100 us interrupt handling

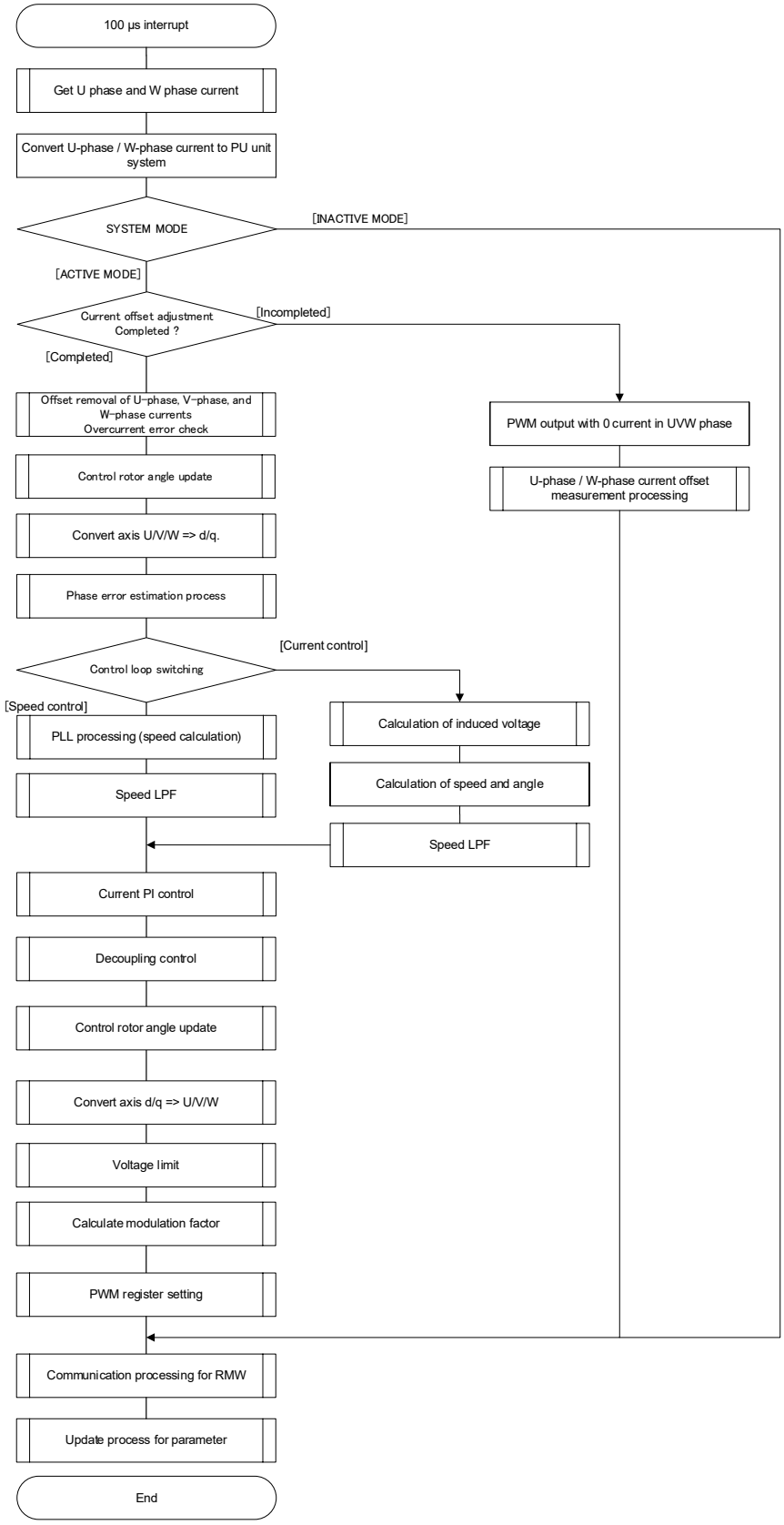

Figure 3-8 Carrier Cycle Interrupt Handling Flowchart

3.6.3 1-ms interrupt handling

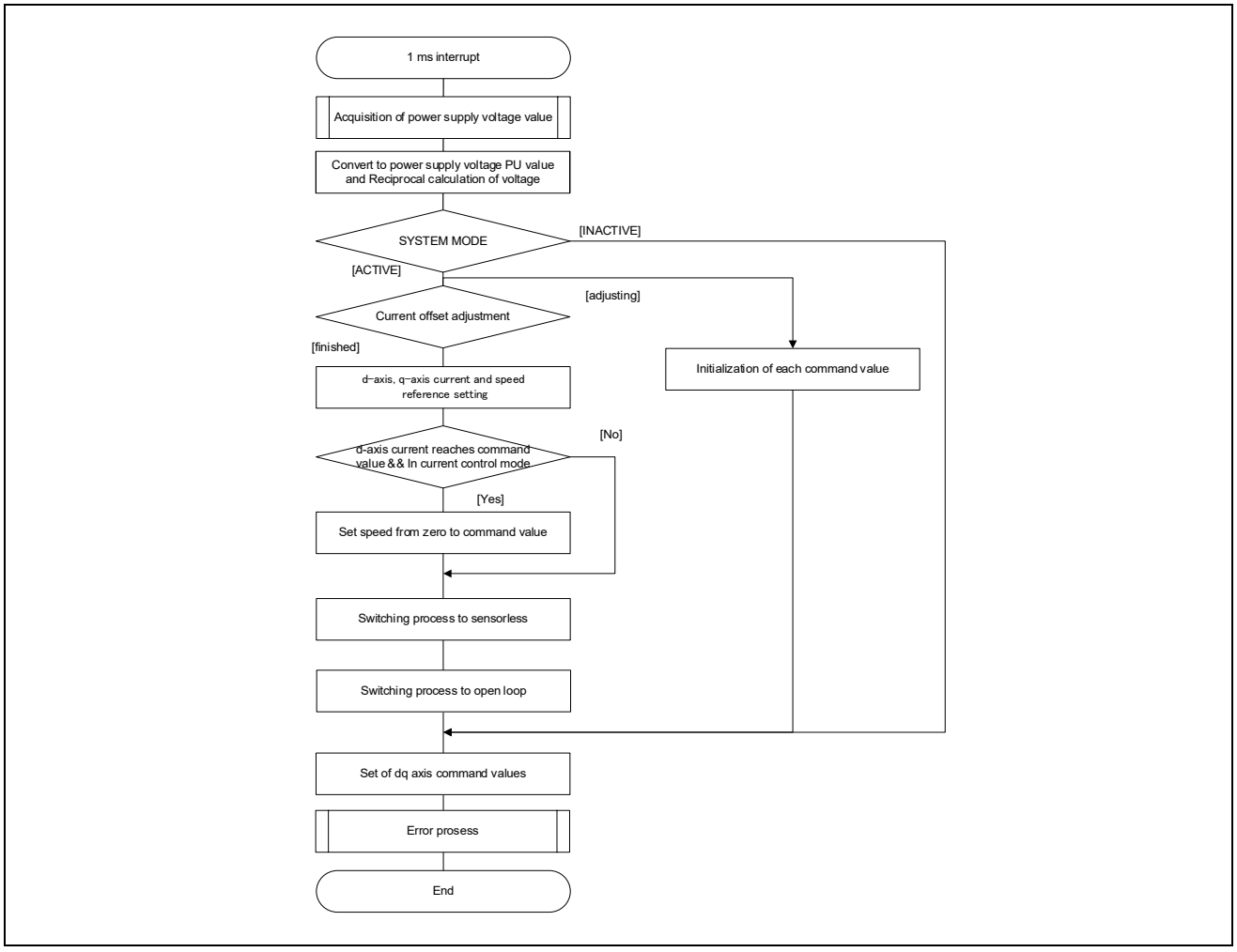

Figure 3-9 1-ms Interrupt Handling Flowchart

3.6.4 Overcurrent interrupt handling

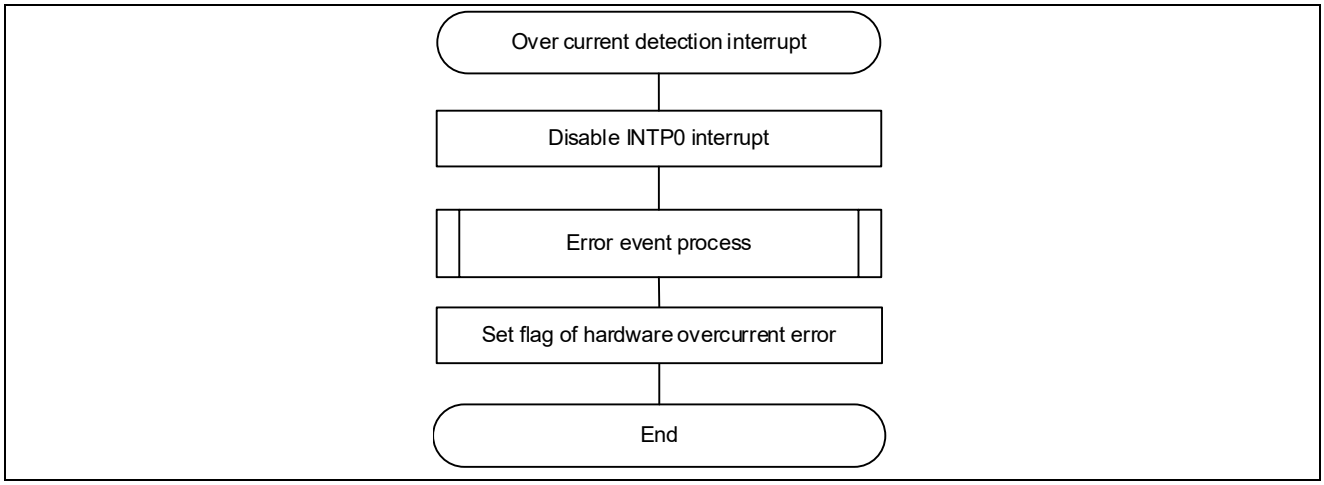

Figure 3-10 Overcurrent Detection Interrupt Handling Flowchart

4. Usage of Motor Control Development Support Tool, Renesas Motor Workbench

4.1 Overview

In the target sample programs described in this application note, you can use user interfaces (rotation/stop command, rotational speed command, etc.) based on the motor control development support tool Renesas Motor Workbench. Please refer to the 'Renesas Motor Workbench V 2.0 User's Manual' for usage and more details. You can find the 'Renesas Motor Workbench' on Renesas Electronics Corporation's website.

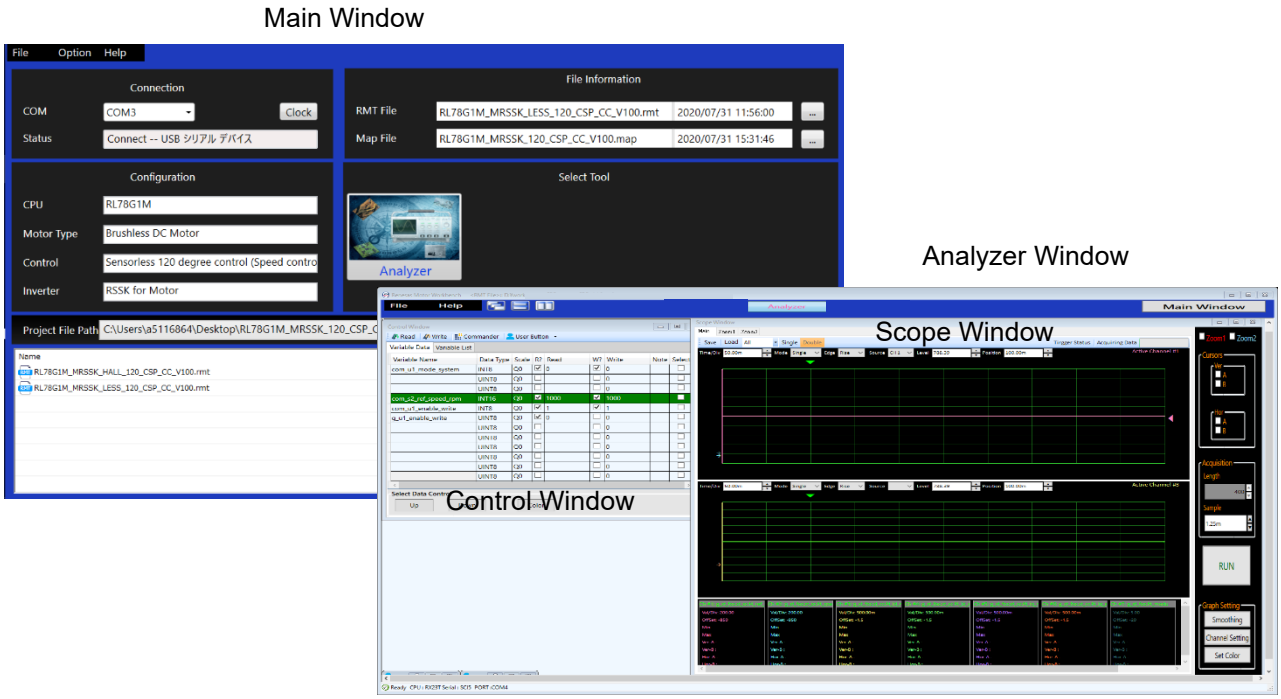

Figure 4-1 Screenshots of Renesas Motor Workbench

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

(1) Start Renesas Motor Workbench by clicking this icon 

.
(2) From the menu bar in the main window, select [File] -> [Open RMT File(O)].
Select RMT file in '[Project Folder]/application/ics/'.
(3) Use the 'Connection' COM select menu to choose the COM port for Motor RSSK.
(4) Click the 'Analyzer' icon on the right side of the Main Window.
(The Analyzer Window will be displayed.)
(5) Please refer to '4.3 Operation Example for Analyzer' for the motor driving operation.

4.2 List of variables for Analyzer

Table 4-1 is a list of variables for the Analyzer. These variable values are reflected to the protect variables when the same values as g_s2_enable_write are written to com_s2_enable_write. However, note that variables with (*) do not depend on com_s2_enable_write.

Table 4-1 List of Input Variables for Analyzer

Variable	Type	Content	Remarks ([ ]: reflection variable name)
com_u1_run_event (*)	uint8_t	Change run mode 0: Stop event 1: Drive event 2: Error event 3: Reset event	[g_u1_run_event]
com_s2_sw_userif (*)	int16_t	Management variable for UI 0: ICS UI (default) 1: Board UI	[g_s2_sw_userif]
com_u1_direction	uint8_t	Direction of rotation 0: CW 1: CCW	[gst_foc.st_asr.u1_ref_dir]
com_f4_mtr_r	float	Resistance [Ω]	[gst_foc.st_motor.s2_mtr_r]
com_f4_mtr_ld	float	d-axis inductance [H]	[gst_foc.st_motor.s2_mtr_ld]
com_f4_mtr_lq	float	q-axis inductance [H]	[gst_foc.st_motor.s2_mtr_lq]
com_f4_mtr_m	float	Induced voltage constant [Vs/rad]	[gst_foc.st_motor.s2_mtr_m]
com_f4_mtr_j	float	Inertia [kgm^2]	[gst_foc.st_motor.s2_mtr_j]
com_u2_mtr_pp	uint16_t	Number of pole pairs	[gst_foc.st_motor.s2_mtr_pp]
com_u2_offset_calc_time	uint16_t	Current offset detection time	[gst_foc.st_tscs.u2_offset_calc_time]
com_s2_ref_speed_rpm	int16_t	Command rotational speed [rpm]	[gst_foc.st_asr.s2_ref_speed_rad]
com_f4_ramp_limit_speed_rpm	float	Limit of acceleration [rpm/ms]	[gst_foc.st_asr.s2_ramp_limit_speed_rad]
com_s2_max_speed_rpm	int16_t	Maximum speed [rpm]	[gst_foc.st_asr.s2_max_speed_rad]
com_f4_acr_nf_hz	float	Current PI control natural frequency [Hz]	[gst_foc.st_acr.st_pi_id.s2_kp] [gst_foc.st_acr.st_pi_id.s2_kidt] [gst_foc.st_acr.st_pi_iq.s2_kp] [gst_foc.st_acr.st_pi_iq.s2_kidt]
com_f4_asr_nf_hz	float	Speed PI control natural frequency [Hz]	[gst_foc.st_asr.st_pi.s2_kp] [gst_foc.st_asr.st_pi.s2_kidt]
com_f4_asr_lpf_cof_hz	float	ASR LPF natural frequency [Hz]	[gst_foc.st_acr.st_iq_lpf.s2_in_k] [gst_foc.st_acr.st_iq_lpf.s2_out_k]
com_f4_pll_nf_hz	float	PLL natural frequency [Hz]	[gst_foc.st_pll.st_pi.s2_kp] [gst_foc.st_pll.st_pi.s2_kidt]
com_s2_less2ol_speed_rpm	int16_t	Switching speed from sensorless to open loop [rpm]	[gst_foc.st_asr.s2_less2ol_speed_rad]
com_s2_ol2less_speed_rpm	int16_t	Switching speed from open loop to sensorless [rpm]	[gst_foc.st_asr.s2_ol2less_speed_rad]
com_f4_ol_ref_id	float	Open loop d-axis command current [A]	[gst_foc.st_acr.s2_ol_ref_id]
com_f4_init_asr_intg	float	ASR integral term initial value during sensorless transition	[gst_foc.st_asr.s2_init_intg]
com_f4_ramp_limit_current	float	Limit value for current rise [A/ms]	[gst_foc.st_acr.s2_ramp_limit_current]
com_s2_enable_write	int16_t	Variable to allow to variable writing	[g_s2_enable_write]

4.3 Operation Example for Analyzer

An example of a motor driving operation using Analyzer is shown below. For the operation, the "Control Window" shown in Figure 4-1 is used. Refer to the 'Renesas Motor Workbench V 2.0 User's Manual' for details about the "Control Window."

- Driving the motor
  - ① Confirm that the [W?] check boxes contain checkmarks for "com_u1_run_event", "com_s2_ref_speed_rpm", and "com_s2_enable_write."
  - ② Input a reference rotational speed value in the [Write] box of "com_s2_ref_speed_rpm."
  - ③ Click the "Write" button.
  - ④ Click the "Read" button. Confirm the [Read] box of "com_s2_ref_speed_rpm" and "g_s2_enable_write."
  - ⑤ Input the value in the [Read] box of "g_s2_enable_write", confirmed in step (4), in the [Write] box of "com_s2_enable_write."
  - ⑥ Input a value of "1" in the [Write] box of "com_u1_run_event."
  - ⑦ Click the "Write" button.

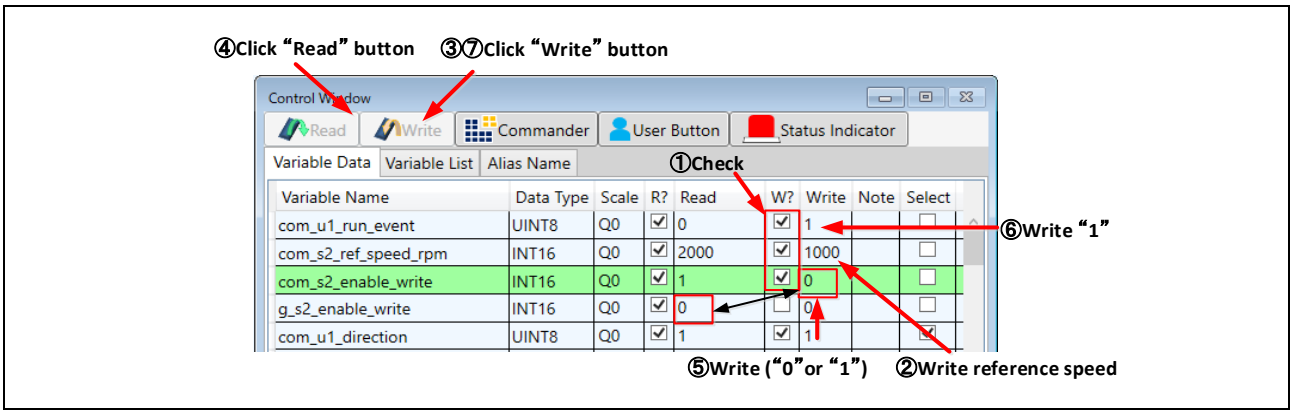

Figure 4-2 Procedure - Driving the motor

- Stop the motor
  - ① Input a value of "0" in the [Write] box of "com_u1_run_event."
  - ② Click the "Write" button.

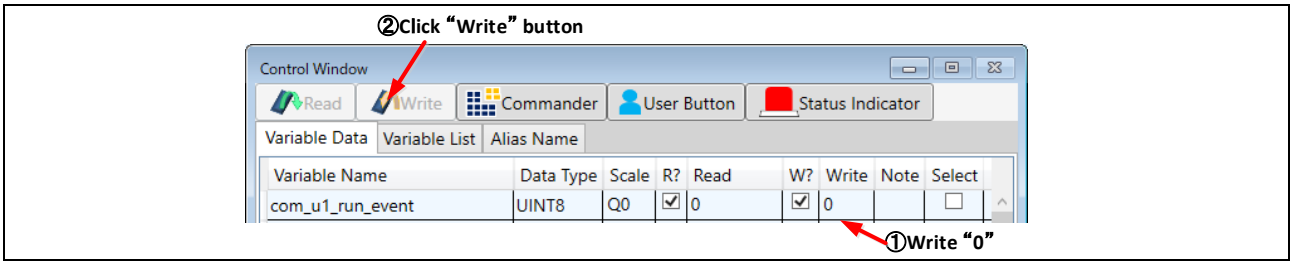

Figure 4-3 Procedure - Stop the motor

- Error cancel operation
  - ① Input a value of "3" in the [Write] box of "com_u1_run_event."
  - ② Click the "Write" button.

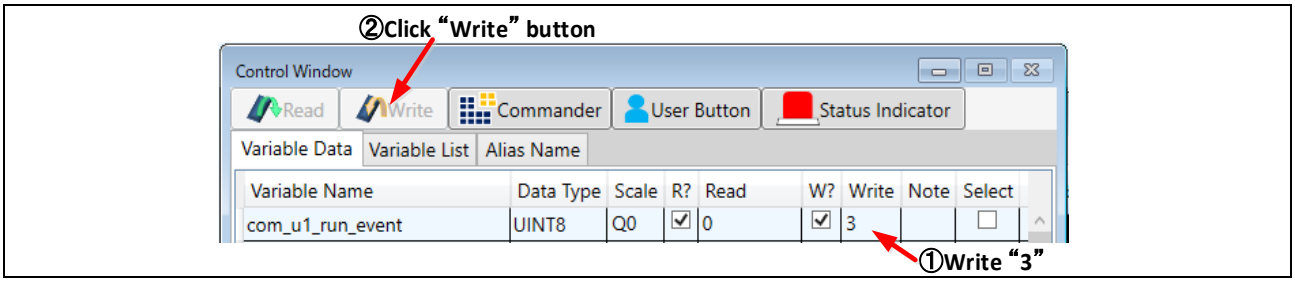

Figure 4-4 Procedure - Error cancel operation

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul, 31, 2017	-	First edition issued
2.00	Oct, 20, 2020	-	Changed motor control algorithm and scaling of the software

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(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.4.0-1 November 2017)

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/.