# RX24T Group

## Vector Control of a Two-Phase Stepping Motor Incorporating a Resolver Sensor

### Introduction

This application note describes the sample software for driving a two-phase stepping motor incorporating a resolver sensor.

### Verified Operation Environment

- MCU: RX24T (R5F524TAADFM)
- Renesas Solution Starter Kit (RSSK): Evaluation System for Stepping Motor with Resolver (RTK0EMX270S01020BJ)
- Motor: Two-phase stepping motor (R17PMK440CNVA4438 manufactured by MinebeaMitsumi Inc.)
- Position sensor: Resolver
- Resolver-to-digital converter (RDC): RAA3064002GFP and RAA3064003GFP
- Motor control development support tool: Renesas Motor Workbench (RMW) 2.0

### Target Sample Program

- RX24T_MRSSK_STM_RSLV_FOC_CSP_RVxxx
- RX24T_MRSSK_STM_RSLV_FOC_E2S_RVxxx

### Reference Documents

- RX24T Group User's Manual: Hardware (R01UH0576EJ0200)
- Renesas CMOS ICs Resolver-to-Digital Converters User's Manual: Hardware (R03UZ0002EJ0100)
- Application Note: Using the Driver for Resolver-to-Digital Converter Control (R03AN0013EJ0010)
- Renesas Motor Workbench 2.0 User's manual (r21uz0004ej0201-motor.pdf)

## Contents

# 1. Overview

This application note describes how to install the software for vector control of a two-phase stepping motor incorporating a resolver sensor using the RX24T microcontroller and how to use the motor control development support tool “Renesas Motor Workbench”.

## 1.1 Development Environment

The following shows the development environment used in this application note.

**Table 1.1 Hardware Development Environment**

| Microcontroller | Evaluation Board | Motor |
|---|---|---|
| RX24T (R5F524TAADFM) | Evaluation System for Stepping Motor with Resolver (RTK0EMX270S01020BJ) | R17PMK440CNVA4438 (incorporating a resolver sensor) |

**Table 1.2 Software Development Environment**

| CS+ Version | e$^2$studio Version | Toolchain Version |
|---|---|---|
| V8.05.00 | 2021-01 | CC-RX V3.03.00 |

# 2. Overview of the System

This section gives an overview of this system.

## 2.1 Hardware Configuration

Figure 2.1 illustrates the hardware configuration.

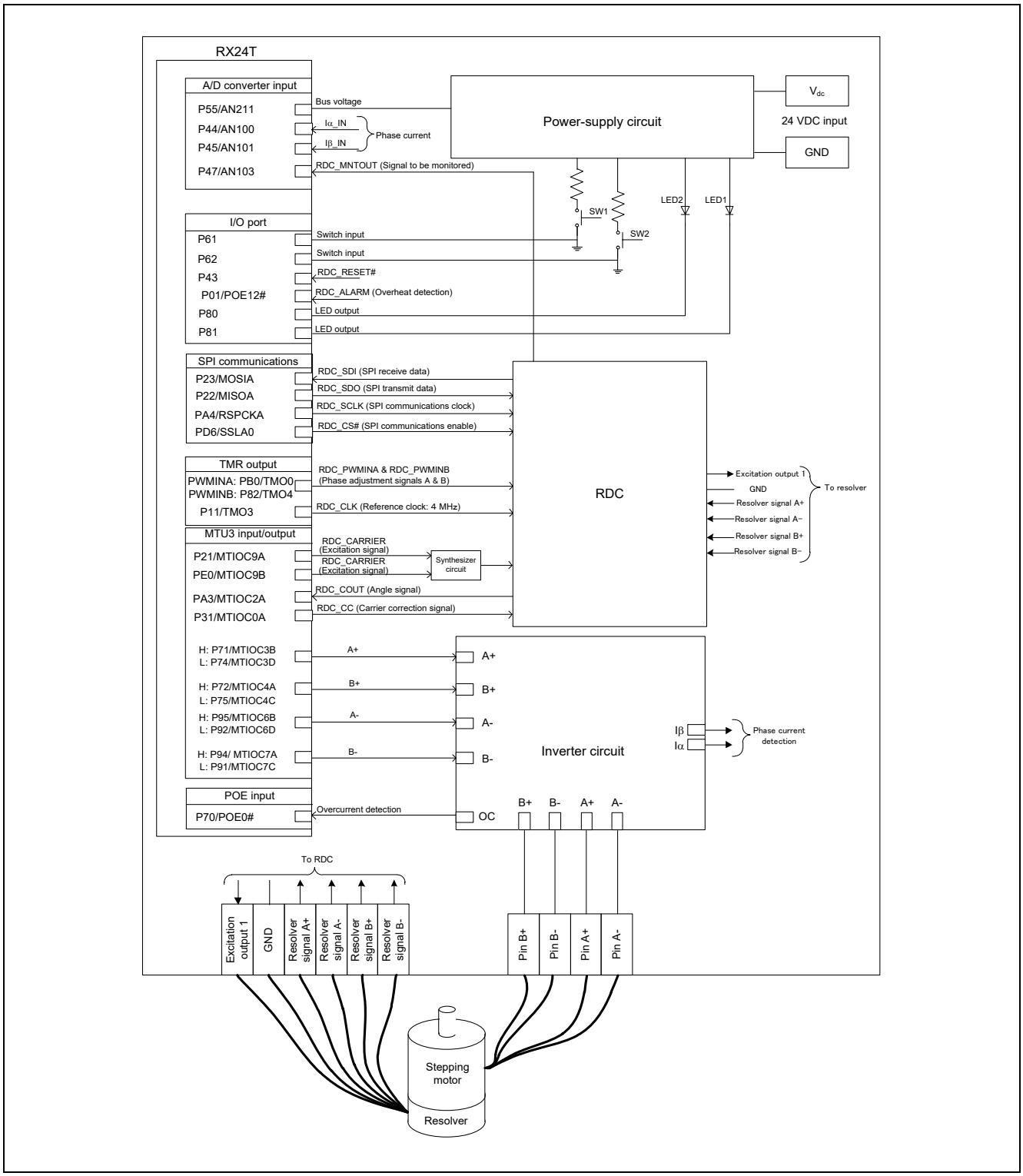



**Figure 2.1 Hardware Configuration**

## 2.2 Hardware Specifications

### 2.2.1 User Interface

Table 2.1 lists the user interfaces of this system.

**Table 2.1 User Interfaces**

| Item | Interface Component | Function |
|---|---|---|
| LED1 | Orange LED (User LED) | Not used |
| LED2 | Orange LED | Not used |
| SW1 | Toggle switch | Turning the motor on and off |
| SW2 | Push switch | Release from the system error states and applying automatic calibration in response to errors |
| VR1 | Variable resistor | Changing the speed reference value |

Table 2.2 shows the pin interfaces of this system.

**Table 2.2 Pin Interfaces**

**R5F524TAADFM**

| Pin Name | Function |
|---|---|
| P55/AN211 | Inverter bus voltage measurement |
| P81 | User LED1 ON/OFF control |
| P80 | User LED2 ON/OFF control |
| P61/IRQ2 | Toggle switch SW1 |
| P62 | Push switch SW2 |
| P01/POE12# | RDC alarm signal (overheat detection) |
| P44/AN100 | Phase α current measurement |
| P45/AN101 | Phase β current measurement |
| P47/AN103 | RDC signal to be monitored |
| P60/AN200 | Variable resistor VR1 |
| P71/MTIOC3B | PWM output (A+ high side) |
| P74/MTIOC3D | PWM output (A+ low side) |
| P95/MTIOC6B | PWM output (A- high side) |
| P92/MTIOC6D | PWM output (A- low side) |
| P72/MTIOC4A | PWM output (B+ high side) |
| P75/MTIOC4C | PWM output (B+ low side) |
| P94/MTIOC7A | PWM output (B- high side) |
| P91/MTIOC7C | PWM output (B- low side) |
| P21/MTIOC9A | Excitation signal output 1 to RDC |
| PE0/MTIOC9B | Excitation signal output 2 to RDC |
| PB0/TMO0 | Resolver phase A adjustment signal output to RDC |
| P82/TMO4 | Resolver phase B adjustment signal output to RDC |
| PA3/MTIOC2A | Angle signal input from RDC |
| P31/MTIOC0A | Carrier correction signal output to RDC |
| P11/TMO3 | Reference clock signal output to RDC |
| P22/MISOA | SPI transmit data to RDC |
| P23/MOSIA | SPI receive data from RDC |
| PA4/RSPCKA | SPI communications clock for RDC |
| PD6/SSLA0 | SPI communications enable signal for RDC |
| P70/POE0# | PWM emergency stop input when an overcurrent is detected |

### 2.2.2 Peripheral Functions for Motor Control

Table 2.3 lists the peripheral functions used in this system.

**Table 2.3 Peripheral Functions**

| Function Name | Description |
|---|---|
| S12ADF | • Phase α and phase β current detection (AN100 and AN101)<br>• Inverter bus voltage detection (AN211) |
| CMT | • 250-μs interval timer (CMT0) |
| MTU3 | • Complementary PWM output (channels 3, 4, 6 and 7 of MTU3) |
| POE3 | • Drives the PWM output pins to high impedance to stop PWM output (POE0#). |

(1) 12-bit A/D converter (S12ADF)
This function measures phase α current ($I_\alpha$), phase β current ($I_\beta$), and inverter bus voltage ($V_{dc}$) in single-scan mode (using hardware triggers). The sample-and-hold function is used to detect phase α current ($I_\alpha$) and phase β current ($I_\beta$).

(2) Compare match timer (CMT)
Channel 0 of the compare match timer is used as a 250-μs interval timer.

(3) Multifunctional timer pulse unit 3 (MTU3)
Channels 3, 4, 6, and 7 of MTU3 are used in the complementary PWM mode to output PWM signals (active high) with dead time for motor control.

(4) Port output enable 3 (POE3)
When an overcurrent is detected (a falling edge on the POE0# pin is detected) or when a PWM output short-circuit is detected, the PWM output pin is driven to high impedance.

### 2.2.3 Peripheral Functions for RDC and Resolver Sensor Control

This sample program uses resolver control driver software (hereafter called the RDC driver) to control the RDC. The RDC driver uses the peripheral functions of the MCU to control the RDC.

**Table 2.4 RDC Driver Functions**

| Function Name | Description |
|---|---|
| S12ADF | • Detects the RDC monitor output signal (AN103). |
| CMT | • Updates the PWM duty cycle of the carrier correction signal (CMT1). |
| MTU3 | • Detects the angle signal (MTU3_2).<br>• Outputs the excitation signal (MTU3_9).<br>• Outputs the carrier correction signal (MTU3_0). |
| TMR | • Outputs the clock for communications between the RDC and SPI module (TMR3).<br>• Outputs the RDC phase adjustment signals (TMR0 and TMR4). |
| IRQ | • Detects RDC alarms (POE12#). |
| RSPI | • Handles communications between the RDC and SPI module (RSPCKA, MISOA, MISIA, and SSLA0). |

(1) 12-bit A/D converter (S12ADF)
- Measures the analog value of the angle signal from the RDC. This function is used for RDC adjustment.

(2) Compare match timer (CMT)
- Updates the duty cycle of the PWM signal used as the carrier correction signal.

(3) Multifunctional timer pulse unit 3 (MTU3)
- Angle signal detection
  Detects the angle signal output from the RDC using the input capture function of the MTU.
- Excitation signal output
  Outputs the excitation signal (rectangular wave) to the RDC.
- Carrier correction signal output
  Outputs a signal for correcting analog errors (first-order distortion) on the resolver signals.

(4) 8-bit timer (TMR)
- RDC reference clock output
  Outputs the reference clock to the RDC.
- RDC phase adjustment signal output
  Outputs a signal for adjusting the phase difference between two-phase resolver signals input to the RDC.

(5) External interrupt requests (IRQ)
- Detects the alarm signal output from the RDC.

(6) Serial peripheral interface (RSPI)
- Handles SPI communications with the RDC.

## 2.3 Software Configuration

### 2.3.1 Folder and File Configuration

The following table shows the configuration of folders and files of the software.

**Table 2.5 Folder and File Configuration (1/2)**

| Folder Hierarchy | | | File | Description |
|---|---|---|---|---|
| **application** | **main** | | main.h, main.c | Main function |
| | **mcu** | | r_mtr_interrupt.c | Interrupt function definitions |
| | **user_interface** | **ics** | r_mtr_ics.c, r_mtr_ics.h | RMW-related function definitions |
| | | | ICS_RX24T.obj | Communications library for RMW |
| | | **flash** | r_mtr_flash.c, r_mtr_flash.h | Flash memory data processing function definitions |
| **r_mtr_spm_rslv_foc_rx** | | | r_mtr_stm_rslv_foc_rx_if.h | Header file for the motor control module |
| | **src** | | r_mtr_stm_rslv_foc_rx.c | User access function definitions |
| | | **mode** | r_mtr_foc_action.c | Action function definitions |
| | | | r_mtr_statemachine.h, r_mtr_statemachine.c | State machine function definitions |
| | | **controller** | r_mtr_common.h | Common definitions |
| | | | r_mtr_foc_stm_rslv_control.h, r_mtr_foc_stm_rslv_control.c | FOC function definitions |
| | | | r_mtr_parameter.h | Parameter definitions |
| | | | r_mtr_ctrl_gain_calc.obj r_mtr_ctrl_gain_calc.h | Control gain calculation function definitions |
| | | | r_mtr_foc_current.h, r_mtr_foc_current.c | Current control function definitions |
| | | | r_mtr_foc_position.h, r_mtr_foc_position.c | Position control function definitions |
| | | | r_mtr_foc_speed.h, r_mtr_foc_speed.c | Speed control function definitions |
| | | | r_mtr_pi_control.h, r_mtr_pi_control.c | PI control function definitions |
| | | | r_mtr_position_profiling.h, r_mtr_position_profiling.c | Position reference value creation function definitions |
| | | | r_mtr_transform.h, r_mtr_transform.c | Coordinate transformation function definitions |
| | | | r_mtr_variables.h, r_mtr_variables.c | Motor control value setting function definitions |
| | | **functions** | r_mtr_volt_err_comp.h, r_mtr_volt_err_comp.obj | Voltage error compensation function definitions |
| | | | r_mtr_speed_observer.h, r_mtr_speed_observer.obj | Speed observer function definitions |
| | | | r_mtr_mod.h, r_mtr_mod.c | Modulation function definitions |
| | | | r_mtr_filter.h, r_mtr_filter.c | General filter function definitions |
| | | | r_mtr_fluxwkn.h r_mtr_fluxwkn.obj | Flux-weakening control |
| | | | r_mtr_ipd.h, r_mtr_ipd.obj | I-PD control function definitions |
| | | | r_rslv_compensation.h, r_rslv_compensation.obj | Resolver angle correction function definitions |
| | | **targets** | | |
| | **ref** | | r_mtr_config.h | Common configuration definitions |
| | | | r_mtr_control_parameter.h | Control parameter configuration definitions |
| | | | r_mtr_inverter_parameter.h | Inverter parameter configuration definitions |
| | | | r_mtr_motor_parameter | Motor parameter configuration definitions |
| | | | r_resolver_rdc_command.h | Initial values of the RDC registers |
| | **lib** | | | |

**Table 2.5 Folder and File Configuration (2/2)**

| Folder Hierarchy | | File | Description |
|---|---|---|---|
| **driver** | **inverter** | r_mtr_ctrl_inverter.h,<br>r_mtr_ctrl_inverter.c | Inverter board-dependent function definitions |
| | **mcu** | r_mtr_ctrl_rx24t.h,<br>r_mtr_ctrl_rx24t.c | MCU-specific function definitions |
| | | r_mtr_ctrl_mcu.h | Common MCU-related definitions |
| | **auto_generation** | (Omitted) | Folder for automatically generated files |
| | **sensor** | | |
| | **rdc_driver_RX** | (Omitted) | RDC driver |

### 2.3.2 Module Configuration

Figure 2.2 shows the configuration of modules of the software.

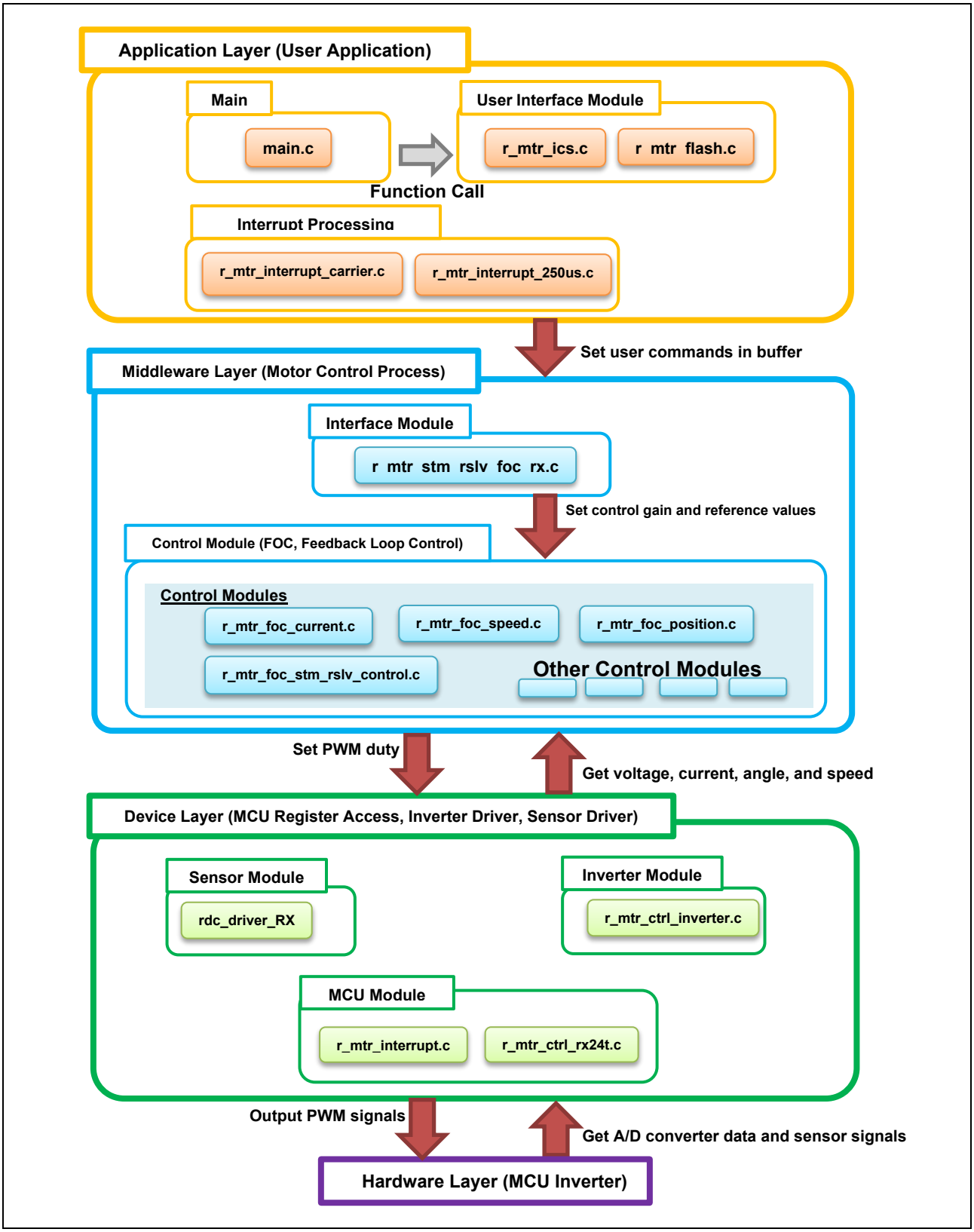



**Figure 2.2 Configuration of the Motor Control Software Modules**

## 2.4 Software Specifications

### 2.4.1 Motor Control

Table 2.6 lists the basic software specifications of this system.

**Table 2.6 Basic Specifications of the Vector Control Software**

| Item | Description |
|---|---|
| Control method | Vector control |
| Motor position control start and stop | Input from the Renesas Motor Workbench (RMW) |
| Rotor's magnetic pole position detection | Resolver sensor (50 poles) |
| Input voltage | 24 VDC |
| Main clock frequency | 80 MHz |
| Carrier frequency (PWM) | 20 kHz (Carrier cycle: 50 µs) |
| Dead time | 0.5 µs |
| Control cycle (current) | 50 µs (Carrier cycle) |
| Control cycle (speed and position) | 250 µs |
| Resolver excitation signal frequency | 20 kHz |
| Resolver angle resolution | 200000 PPR (0.0018 degrees)<br>(Main clock frequency) / (excitation frequency) * (number of poles of resolver sensor) |
| Position reference value management | Creation of position reference value: Position profiling using the trapezoidal speed wave method<br>Input range:<br>-32768° to 32767°<br>Speed limit:<br>Clockwise (CW) and counterclockwise (CCW): 0 to 1500 rpm |
| Speed reference value management | CW: 0 rpm to 2000 rpm<br>CCW: 0 rpm to 2000 rpm |
| Dead band of position* | ±1 count |
| Natural frequency of each control loop | Current control loop: 400 Hz<br>Speed control loop: 40 Hz<br>Position control loop: 10 Hz |
| Compiler optimization settings | Optimization level: 2 (-optimize = 2) (Default setting)<br>Optimizing method: Optimization with the code size precedence (-size) (Default setting) |
| ROM and RAM size | ROM: 66 Kbytes<br>RAM: 6 Kbytes |
| Protection stop processing | The six motor control signal outputs are disabled when any of the following conditions is met.<br>1. The current of a phase exceeds 5 A. (Monitored at intervals of 50 µs)<br>2. The inverter bus voltage exceeds 40 V. (Monitored at intervals of 50 µs)<br>3. The inverter bus voltage is lower than 20 V. (Monitored at intervals of 50 µs)<br>4. The rotational speed exceeds 4000 rpm. (Monitored at intervals of 50 µs)<br>5. An interrupt of the alarm signal detection is issued from the RDC.<br><br>When the overcurrent detection signal (detection of a falling edge on the POE0# pin) from the external circuit or a short circuit of an output is detected, the PWM output pin is driven to high impedance. |

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

### 2.4.2 RDC and Resolver Sensor Control

Table 2.7 lists the settings of the RDC driver. For details, see the application note "Using the Driver for Resolver-to-Digital Converter Control".

**Table 2.7 Settings of the RDC Driver**

| Item | Description |
|---|---|
| Excitation signal output | Output frequency: 20 kHz |
| Carrier correction signal output | Output frequency: 200 kHz |
| Phase adjustment signal output | Output frequency: 400 kHz<br>Duty cycle: Set by the adjustment function (Initial value: 50% for both phase A and phase B) |
| Angle signal input | Captured at an input capture interrupt generated on only rising edges or both rising and falling edges of the angle signal) |
| RDC clock output | Output frequency: 4 MHz |
| Interrupt for updating the duty cycle of the carrier correction signal | Update cycle: 40 kHz |

### 2.4.3 Interrupt Processing

Table 2.8 lists interrupts used in this program.

**Table 2.8 List of Interrupts**

| Interrupt | Purpose of Use | Interrupt Level |
|---|---|---|
| ICU: IRQ5 | RDC alarm detection | 15 |
| POE: OEI1 | Overcurrent detection by an external circuit | 15 |
| CMT1: CMI1 | Update of the duty cycle for carrier correction PWM signal | 14 |
| MTU2: TGIA2 | Angle signal detection (input capture) | 13 |
| MTU4: TCIV4 | Acquisition of angle information at the time of current detection | 12 |
| MTU9: TGIC9 | Synchronization of the excitation signal and carrier correction signal | 11 |
| S12AD: S12ADI | Current control | 10 |
| CMT0: CMI0 | Speed and position control | 9 |
| RSPI0: XXXX * | RDC communications processing | 8 |
| SCI5: RXI5 | RMW communications processing | 6 |

Note: * Although multiple interrupts are used for communications with the RDC (including transmission end, reception end, and error interrupts), a single interrupt level is assigned to all of them.

# 3. Details of the Software

This section describes the software covered by this application note.

## 3.1 Detecting Inverter Bus Voltage and Motor Phase Currents (A/D Conversion)

(1) Inverter bus voltage

The A/D conversion ratio of the inverter bus voltage is shown in the following table. The converted value is used to calculate the rate of PWM signal modulation and detect a low-voltage and an overvoltage. (The PWM output stops when an error is detected.)

**Table 3.1 Conversion Ratio of Inverter Bus Voltage**

| Item | Conversion Ratio (Inverter bus voltage to A/D-converted value) | Channel |
|---|---|---|
| Inverter bus voltage | 0 V to 60 V converted to 0000H to 0FFFH | AN206 |

(2) Phase α and phase β currents

The A/D conversion ratio of the phase α current and phase β current is shown in the following table. The converted values are used for current control (vector control).

**Table 3.2 Conversion Ratio of Phase α and Phase β Currents**

| Item | Conversion Ratio (Phase α or phase β current to A/D-converted value) | Channel |
|---|---|---|
| Phase α and phase β currents | -6.25 A to 6.25 A converted to 0000H to 0FFFH * | Ia: AN100<br>Ib: AN101 |

Note: * For details of A/D conversion characteristics, see *RX24T Group User's Manual: Hardware*.

## 3.2 Controlling the Operating State of the Motor

Figure 3.1 illustrates the operating state of the motor controlled by this sample software. This sample software controls the operating state of the motor with the system mode, run mode, and loop mode.

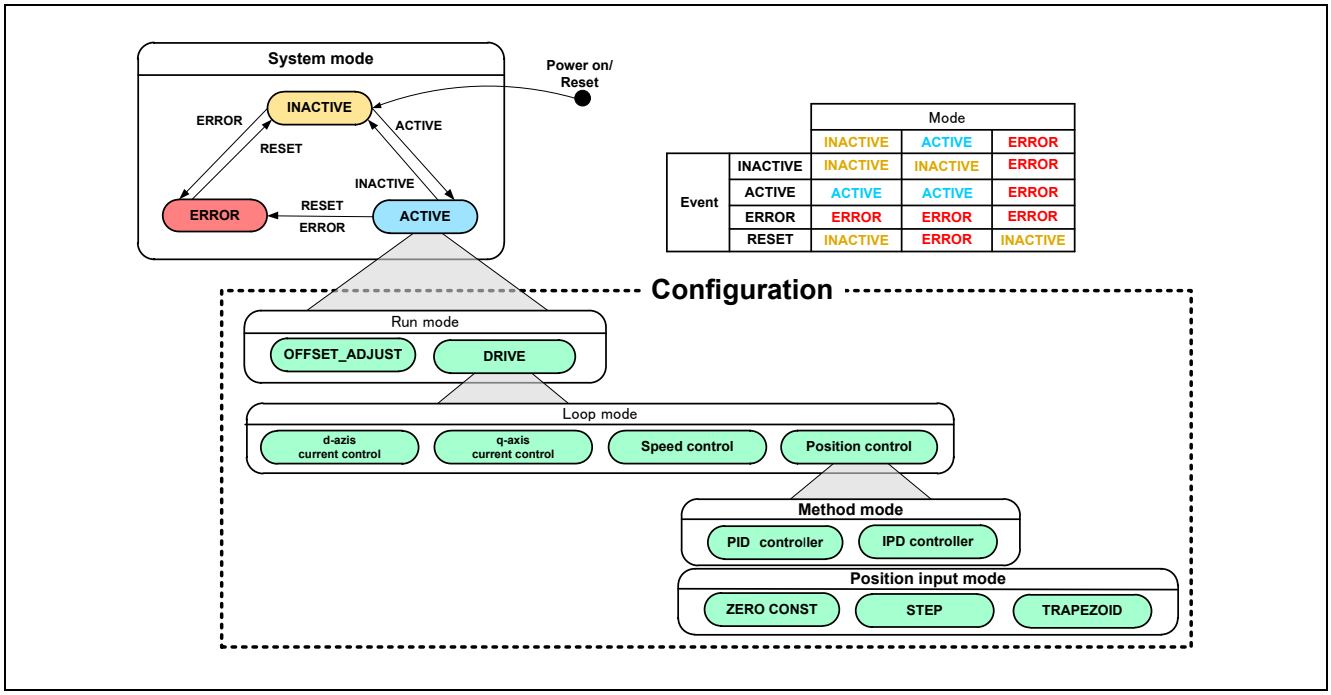



**Figure 3.1 State Transitions of the Stepping Motor Incorporating a Resolver Sensor Controlled by this Software**

(1) System mode

This mode refers to the operating state of the system (motor). A state changes when an event occurs. The state of the system can be INACTIVE (the motor is stopped), ACTIVE (the motor is operating), or ERROR (the motor operation is faulty).

(2) Event

When an event occurs in a state in the system mode, the the operating state of the system changes as shown in Figure 3.1 in accord with the event. The sources of the events are listed in the following table.

**Table 3.3 List of Events**

| Event Name | Description | Source |
|---|---|---|
| INACTIVE | The PWM output is disabled. | User operation* |
| ACTIVE | The PWM output is enabled. | User operation* |
| ERROR | The motor control is deactivated.<br>The PWM output is stopped (disabled).<br>The error state is reflected. | On detection of an error by the system |
| RESET | The error state is cleared.<br>The system is placed in the inactive state and the variables are initialized. | User operation* |

Note: * This event is generated by modifying the value of the variable com_u1_mode_system.

(3) Run mode

This mode refers to the state of the motor control. When the system is placed in ACTIVE, the motor operates in the specified mode.

**Table 3.4 States of the Run Mode**

| State Name | Description | Source |
|---|---|---|
| MTR_MODE_OFFSET_ADJUST | The current and angle offset removal is handled. Once completed, the system is placed in INACTIVE. | User operation* |
| MTR_MODE_DRIVE | The operation specified by the loop mode is handled. | User operation* |

Note: * Transition is triggered by modifying the value of the variable com_u2_run_mode.

(4) Loop mode

When DRIVE is specified in the run mode, the motor is driven as specified in this mode.

**Table 3.5 States of the Loop Mode**

| State Name | Description | Source |
|---|---|---|
| MTR_LOOP_CURRENT_ID | Current of the d axis is manually controlled. | User operation* |
| MTR_LOOP_CURRENT_IQ | Current of the q axis is manually controlled. | User operation* |
| MTR_LOOP_SPEED | Speed is manually controlled. | User operation* |
| MTR_LOOP_POSITION | Position is manually controlled. | User operation* |

Note: * Transition is triggered by modifying the value of the variable com_u1_ctrl_loop_mode.

(5) Method mode

When DRIVE and POSITION CONTROL are respectively specified in the run mode and loop mode, the motor is driven as specified in this mode.

**Table 3.6 States of the Method Mode**

| State Name | Description | Source |
|---|---|---|
| MTR_CTRL_PID | Position is controlled by the PID controller. | User operation* |
| MTR_CTRL_IPD | Position is controlled by the IPD controller. | User operation* |

Note: * Transition is triggered by modifying the value of the variable com_u1_ctrl_method_mode.

(6) Position input mode

When DRIVE and POSITION CONTROL are respectively specified in the run mode and loop mode, the method for inputting the position reference value can be specified in this mode.

**Table 3.7 States of the Position Input Mode**

| State Name | Description | Source |
|---|---|---|
| MTR_POS_ZERO_CONST | Position reference 0 | The control of position is not specified in the loop mode. |
| MTR_POS_STEP | A step response to the input reference value is made. | User operation* |
| MTR_POS_TRAPEZOID | The position profiling is handled to limit acceleration. | User operation* |

Note: * Transition is triggered by modifying the value of the variable com_u1_position_input_mode.

## 3.3 Detecting Rotor Position

The rotor position is detected using the following algorithm.

- Peripheral functions in use
  Channel 9 of MTU3 (MTU3_9): Resolver excitation signal output (PWM output)
  Channel 2 of MTU3 (MTU3_2): Angle signal detection counter (input capture)

1. MTU3_9 starts counting at the excitation frequency (20 kHz) set by the RDC driver in the initialization processing. MTU3_2 starts counting in synchronization with the excitation signal with the same cycle as MTU3_9.
2. The value of the MTU3_2 counter is updated on the rising edges (used as input capture triggers) of the angle signal (input to the PA3/MTIOC2A pin for the program covered by this application note) from the RDC.
   Input capture can also be made on both rising and falling edges of the angle signal.
3. An API function of the RDC driver is used to acquire the MTU3_2 counter value as angle information for a single pole of the resolver sensor. At the same time, the difference between this value and the previously captured counter value is calculated. If the result is greater than 1/2 of the maximum MTU3_2 counter value (or has a value below –1/2 of the maximum counter value in the case of counterclockwise rotation), the counter is judged to have wrapped around and the captured counter value is normalized by adding or subtracting the maximum counter value. The angle is calculated from the normalized value.
   The rotational speed of the rotor can be calculated from the count difference information.

Schematic diagrams are shown below.

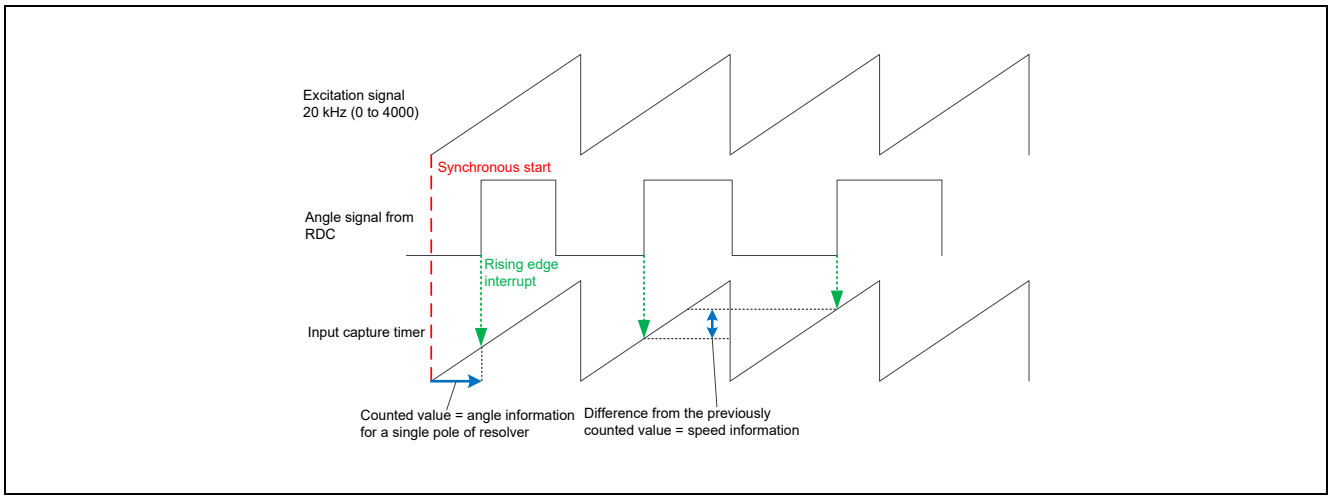



**Figure 3.2 Schematic Diagram of Detecting Rotor Position (on Rising Edges)**

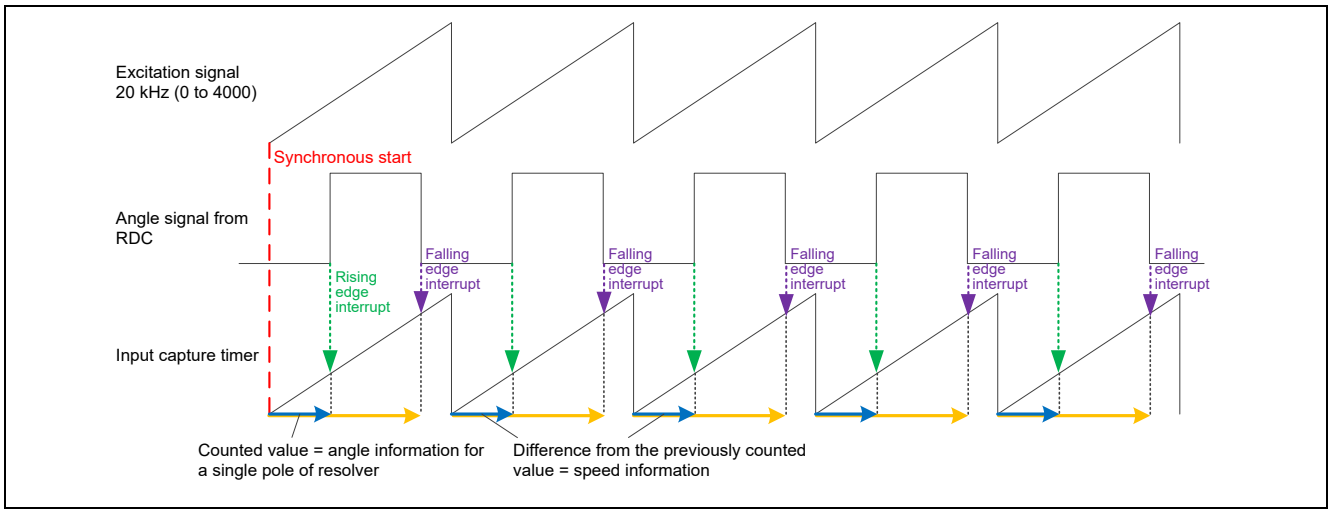



**Figure 3.3 Schematic Diagram of Detecting Rotor Position (on Both Rising and Falling Edges)**

### 3.3.1 Angle Correction at the Time of Current Detection

The angle information obtained from the RDC is updated in response to each input capture interrupt. However, obtaining the angle information at the time of current detection is desirable for current control in motor control. Therefore, it is necessary to correct the angle information to include the amount of change in the angle from the angle information being updated in response to an input capture interrupt and until detection of a current.

This sample software uses MTU3 for excitation signal output (MTU3_9), input capture (MTU3_2), and complementary PWM output for motor control (MTU3_3, MTU3_4, MTU3_6, and MTU3_7). All counters for these signals can be started simultaneously by using an API function of the RDC driver.

Current detection proceeds at the troughs of the complementary PWM timer counter for motor control. As the cycle of this timer is the same as that of the input capture timer, the time elapsed until the trough of the counter is obtained from the counted value that is detected at the time of an input capture interrupt. The correction angle used at current detection can be obtained from this elapsed time and the motor driving speed.

A schematic diagram is shown below.

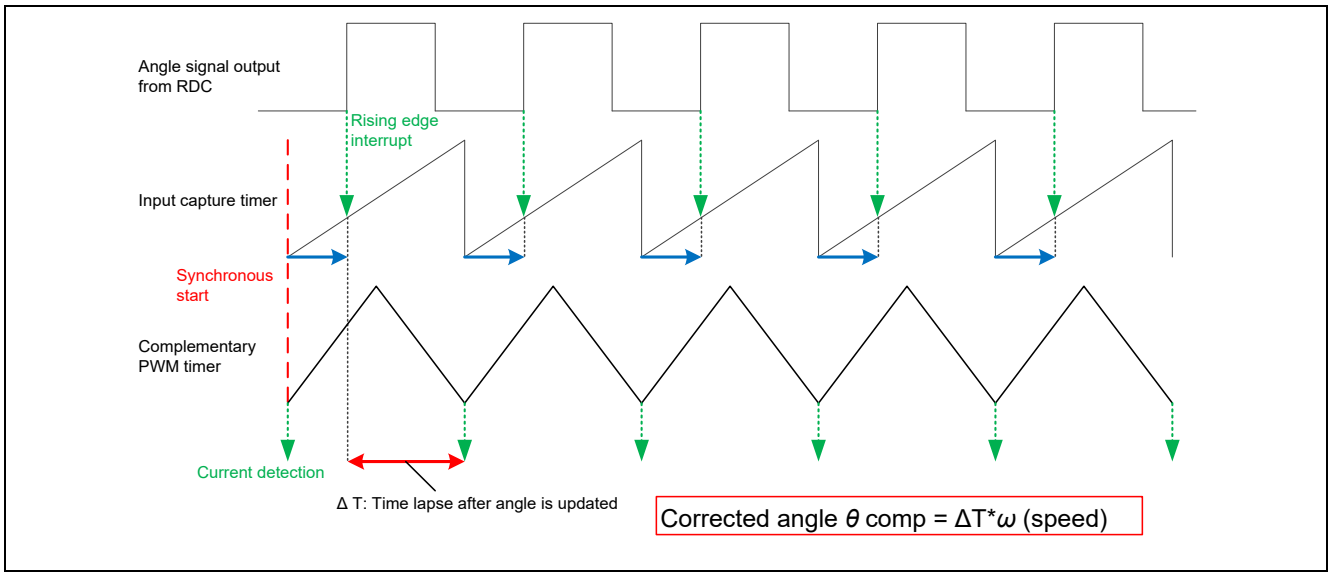



**Figure 3.4 Schematic Diagram of Correcting Angle at the Time of Current Detection**

### 3.3.2 Angle Correction at the Time of Output Voltage Updating

There is a delay from the time of current detection until calculation of the corrected angle and reflection of the reference voltage in the PWM signal. Therefore, the angle must be corrected considering this delay. The angle at the time of current detection is used for αβ-to-dq coordinate transformation and the angle obtained by correcting the delay is used for dq-to-αβ coordinate transformation.

A schematic diagram is shown below.

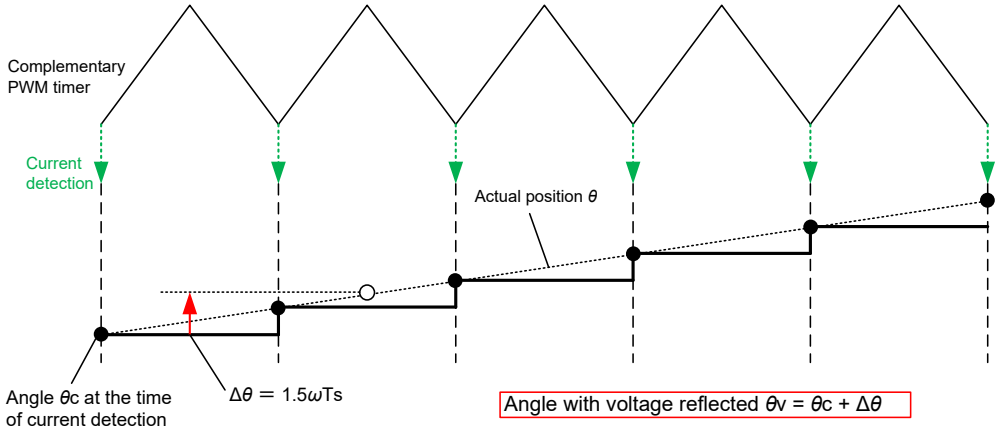



**Figure 3.5 Schematic Diagram of Correcting Angle at the Time of Output Voltage Updating**

## 3.4 Removing Offset of Sensor Position against Magnetic Pole Position

When using a resolver sensor, the magnetic pole position must be aligned with the reference angle of the sensor. The orientation of the current vector is aligned with the d axis to determine the initial magnetic pole position according to the procedure shown in Figure 3.6. This angle is stored as an offset value and is subtracted from the angles detected during motor control. Figure 3.7 shows the starting sequence at this time.


(Step 1) OFFSET ADJUST mode

A current vector is created in the direction perpendicular to the α-axis.

(Step 2) OFFSET ADJUST mode

A current vector is created in the α-axis direction.

**Figure 3.6 Determination of Permanent Magnet Position**

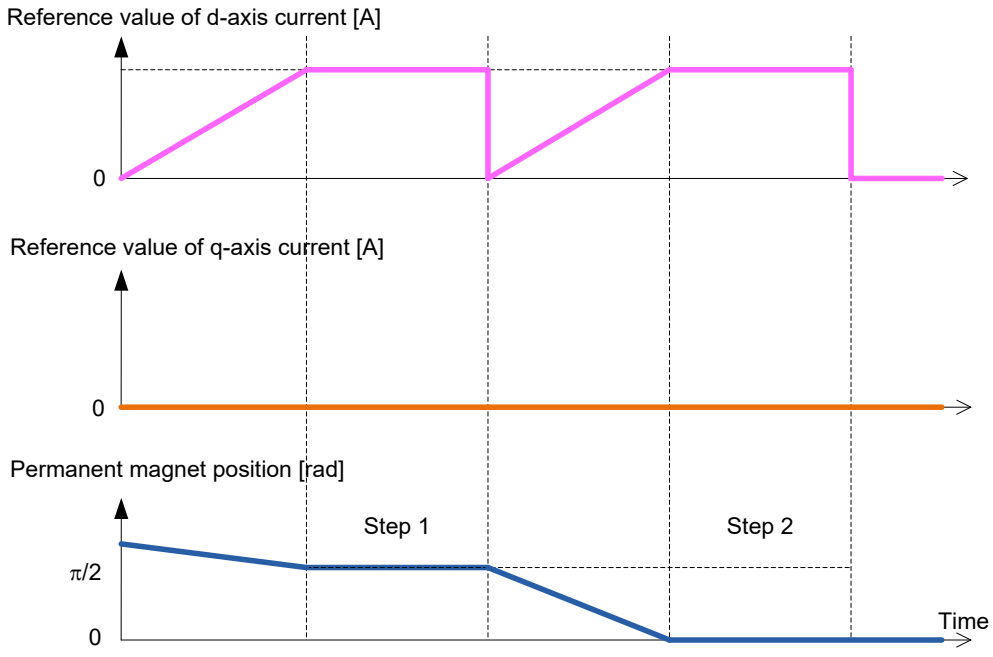
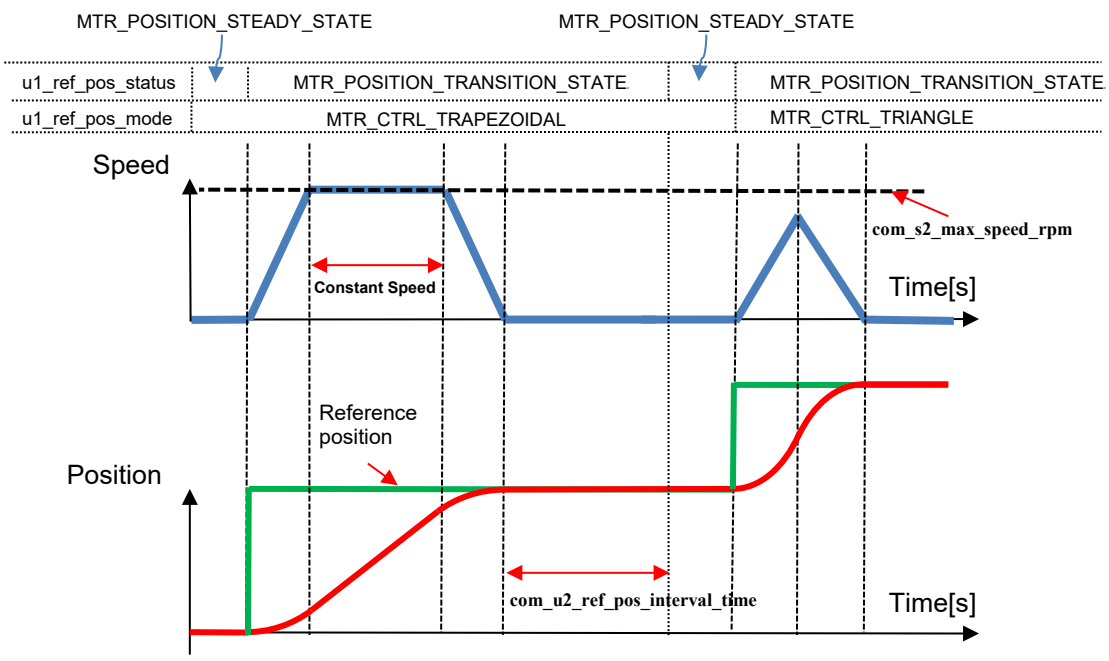



**Figure 3.7 Example of Offset Removing Sequence**

## 3.5 Creating a Position Reference Value (Moving Average-Type Acceleration and Deceleration Method)

Position control provides a function for re-calculating the position reference value according to the specified position reference value, acceleration or deceleration time, and maximum speed in every control cycle to control the speed reference value. The following diagram gives an overview of this function




MTR_CTRL_TRIANGLE: f4_accel_max_ref_speed_rad * f4_accel_time >= f4_dt_pos_rad

MTR_CTRL_TRAPEZOIDAL: f4_accel_max_ref_speed_rad * f4_accel_time < f4_dt_pos_rad

**Figure 3.8 Shaping of Position Reference Value and Triangular or Trapezoidal Speed Reference Value**

The following values can be specified in the RMW to create a position reference value with the acceleration or deceleration time taken into account.

- Acceleration time (com_f4_accel_time)
- Maximum speed (com_s2_max_speed_rpm)
- Settling wait time (com_u2_ref_pos_interval_time)

If the speed value obtained from the position difference and acceleration time is greater than the maximum accelerating speed, a position reference value is created so that the speed reference value becomes trapezoidal.

## 3.6 Vector Control Algorithm

### 3.6.1 Analysis Model of Two-Phase Stepping Motor

The voltage equation of a two-phase stepping motor is shown below.

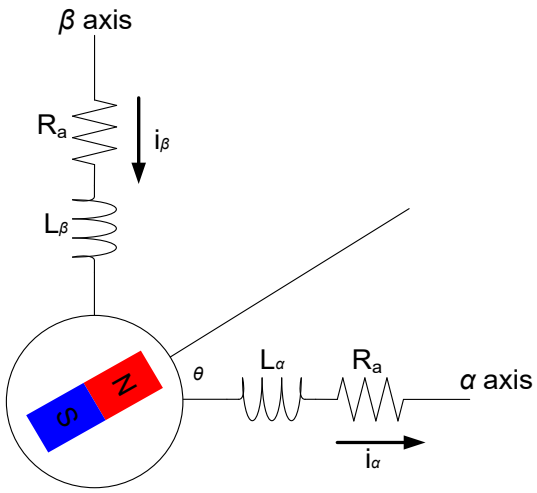



**Figure 3.9 Schematic Diagram of Two-Phase Stepping Motor**

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} R_a + pL_\alpha & pL_{\alpha\beta} \\ pL_{\alpha\beta} & R_a + pL_\beta \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \omega\psi \begin{bmatrix} -sin\theta \\ cos\theta \end{bmatrix}$$

$v_\alpha, v_\beta$: α-axis and β-axis armature voltages

$i_\alpha, i_\beta$: α-axis and β-axis armature currents

$R_a$: Armature resistance of each phase

$\omega$: Angular speed

$p$: Differential operator

$\psi$: Maximum value of interlinkage magnetic flux of armature due to permanent magnet

$\theta$: Lead angle of permanent magnet (rotor) from phase α

$L_\alpha, L_\beta, L_{\alpha\beta}$: Inductances

### 3.6.2 d-q-Axis Model of Two-Phase Motor

In vector control, an AC two-phase (α, β) coordinate system is represented by a DC two-phase (d, q) coordinate system. Because the two-phase coils of the stator are converted to two-phase coils that rotate in synchronization with the rotor of the permanent magnet, the two-phase coils are stationary relative to the rotor and are handled as two electrically independent DC circuits.

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

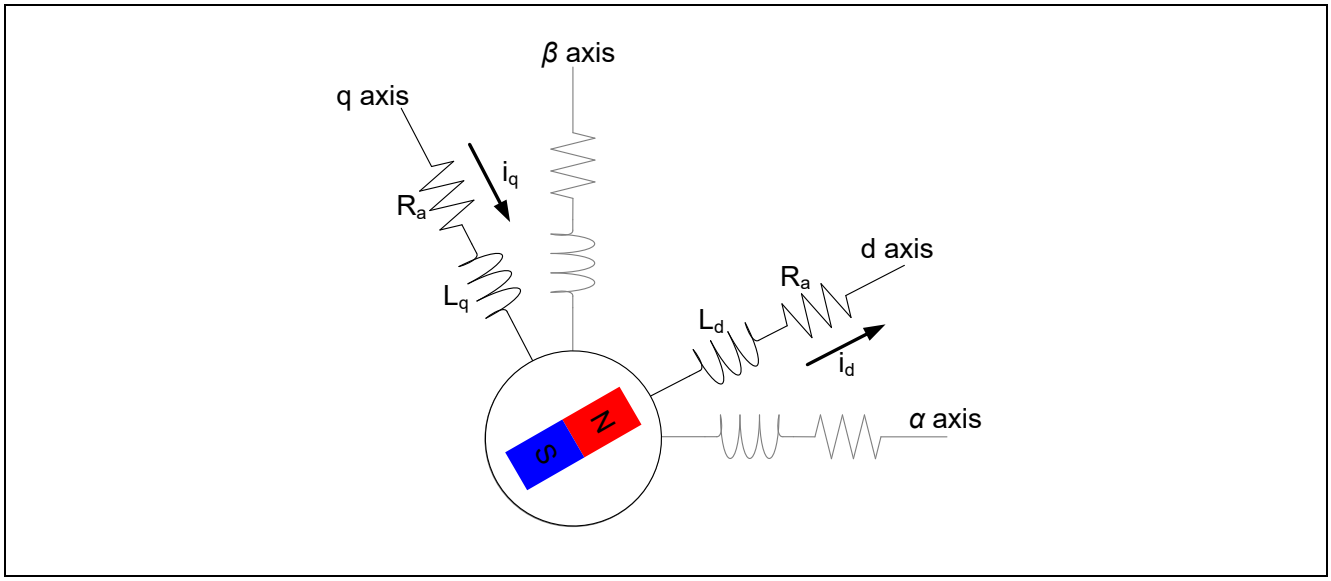



**Figure 3.10 Schematic Diagram of Two-Phase DC Motor**

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

Based on the above coordinate transformation, the voltage equation in the dq coordinate system is shown below.

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

$v_d, v_q$: d-axis and q-axis armature voltages

$i_d, i_q$: d-axis and q-axis armature currents

$R_a$: Armature resistance of each phase

$\omega$: Angular speed

$p$: Differential operator

$L_d, L_q$: d-axis and q-axis inductances

$\psi_a$: Effective value of interlinkage magnetic flux of armature due to permanent magnet

Accordingly, the alternate currents flowing through the stationary two-phase stator can be regarded as direct currents that flow through the two-phase stator rotating in synchronization with the permanent magnet (rotor).

### 3.6.3 Torque Generated in a Motor

The magnitude of torque generated in a motor is obtained by the following equation using the outer product of the current vector and the armature's interlinkage magnetic flux. The first member of the right side of this equation is called the magnet torque and the second member of the right side is called the reluctance torque.

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

$T$: Motor torque $P_n$: Number of pole pairs

A motor with no difference between the d-axis inductance and the q-axis inductance is called a motor without saliency. In this case, the torque increases in proportion to the q-axis current because of a zero reluctance torque. Therefore, the q-axis current is called the torque current in some cases. On the other hand, the d-axis current is called the excitation current in some cases because varying this current makes the apparent size of the magnetic flux of the permanent magnet change.

## 3.7 Vector Control System and Controllers

Figure 3.11 shows a block diagram of the overall position control system.

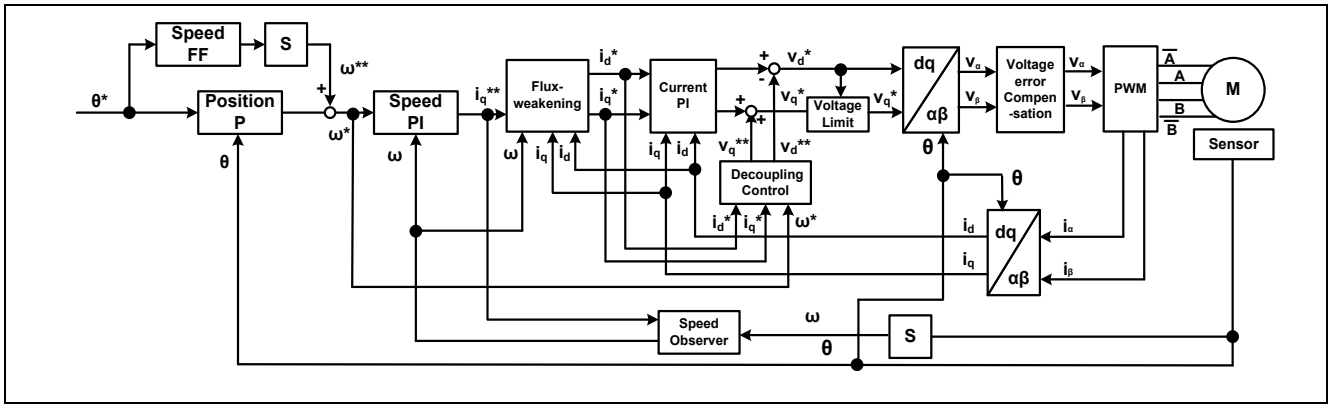



**Figure 3.11 Block Diagram of the Vector Control System (Position Control)**

As shown in Figure 3.11, the position control system consists of a position control loop, a speed control loop, and a current control loop. The speed control loop and the current control loop are implemented by general PI controllers and the position control loop is implemented by a P controller and a feedforward controller for speed. The gains of respective controllers must be properly designed to achieve desired control characteristics.

In the system block diagram, the decoupling control block feeds forward the inductive voltages $v_d^{**}$ and $v_q^{**}$ (shown in the equations below) generated by motor rotation to the reference voltage of each phase. This achieves high responsiveness in the speed control system and independent control of the d axis and q axis.

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

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

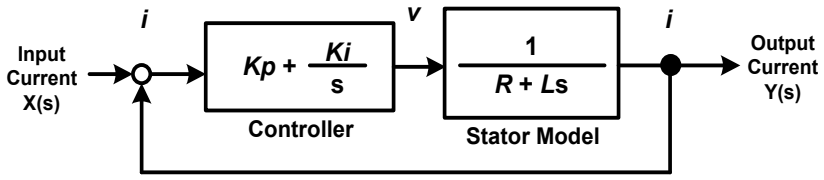
**Table 3.8 Motor Parameter Settings**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| R | com_f4_mtr_r | Measured parameter of motor (resistance) (ohm) |
| $L_d$ | com_f4_mtr_ld | Measured parameter of motor (d-axis inductance) (H) |
| $L_q$ | com_f4_mtr_lq | Measured parameter of motor (q-axis inductance) (H)<br>If this value is smaller than $L_d$, the same value as $L_d$ is set. |
| $\psi_a$ | com_f4_mtr_m | Measured parameter of motor |

### 3.7.1 Designing a Current Control Loop

The current control loop is modeled by using the electrical characteristics of the motor. The stator coil can be represented by a resistance R and an inductance L. So the stator model of the motor is expressed by the transfer function of the typical RL series circuit $\frac{1}{R+Ls}$.

Using a PI controller, the current control loop can be represented as the feedback control loop shown in Figure 3.12.




$R$: Resistance of stator coils $(\Omega)$ $L$: Inductance of stator coils $(\mathrm{H})$

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

**Figure 3.12 Model of the Current Control Loop**

Based on this model, the PI gains of the current control loop are designed using the known $R$ and $L$ values of the motor stator.

The closed-loop transfer function of the current control loop is obtained by the following equation.

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

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

The general formula of the second-order delay system with a zero point can be expressed as follows.

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

The relations expressed by the following equations can be obtained by comparing the coefficients of the transfer function of the current control loop and the general formula of the second-order delay system with a zero point.

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

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

Thus, the natural frequency $\omega_n$, attenuation coefficient $\zeta$, and zero-point frequency $\omega_z$ can be expressed as follows.

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

From these, the current PI control gains $K_{p\_current}$ and $K_{i\_current}$ are obtained as follows.

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

$\omega_{CG}$: Natural frequency of current control loop

$\zeta_{CG}$: Attenuation coefficient of current control loop

Accordingly, the PI control gain of the current control loop can be designed using the $\omega_{CG}$ and $\zeta_{CG}$ values.

**Table 3.9 Settings of the Current Control Loop**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $\omega_{CG}$ | com_f4_current_omega | This parameter is the natural frequency (Hz) of the current control loop. It can be set at up to approximately 1/10 of the current control frequency and in proportion to how often current control is applied. However, a relatively low parameter value is usually set in consideration of noise in position detection and current detection. |
| $\zeta_{CG}$ | com_f4_current_zeta | This parameter is the attenuation coefficient of the current control loop. It is usually set to 0.7 to 1.0. A value closer to 1 enables a stable and gentle response. |

### 3.7.2 Designing a Speed Control Loop

The speed control loop can be modeled by using the mechanical characteristics of the motor. From the rotational motion equation, the mechanical torque can be expressed by the following equation.

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

$J$: Rotor inertia $\omega_{mech}$: Mechanical angular speed

In consideration of only magnet torque, the electrical system torque equation is written as follows.

$$T = P_n \psi_a i_q$$

From the mechanical torque and electrical torque equations, the mechanical angular speed can be expressed as follows.

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

Because an electrical angle is used to handle the speed in the control software, multiply both sides of the equation by the number of pole pairs $P_n$.

$$\omega_{elec} = \frac{{P_n}^2 \psi_a}{sJ} i_q$$

$\omega_{elec}$: Electrical angular speed

This equation is a motor model of the speed control loop. Using a PI controller, the speed control loop can be represented as the feedback control loop shown in Figure 3.13.

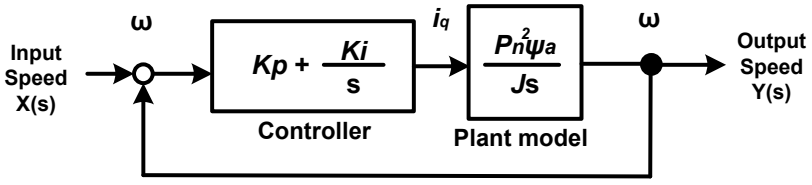



**Figure 3.13 Model of the Speed Control Loop**

Then, specify the PI control gain of the speed control loop assuming that the motor parameters $P_n$, $\psi$, and $J$ are known. Obtain the transfer function of the system first.

The closed-loop transfer function of the speed control loop is obtained by the following equation.

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

$$K_b = \frac{K_P {P_n}^2 \psi}{J}, \quad K_i = K_p a$$

The general formula of the second-order delay system with a zero point can be expressed as follows.

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

As with the current control loop, the relations expressed by the following equations can be obtained by comparing the coefficients of the transfer function of the speed control loop and the general formula of the second-order delay system with a zero point.

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

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

Thus, the natural frequency $\omega_n$, attenuation coefficient $\zeta$, and zero-point frequency $\omega_z$ can be expressed as follows.

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

From these, the PI control gains $K_{p\_speed}$ and $K_{i\_speed}$ are obtained as follows.

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

$\omega_{SG}$: Natural frequency of speed control loop

$\zeta_{SG}$: Attenuation coefficient of speed control loop

Accordingly, the PI control gain of the speed control loop can be designed using the $\omega_{SG}$ and $\zeta_{SG}$ values.

**Table 3.10 Settings of the Speed Control Loop**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $\omega_{SG}$ | com_f4_speed_omega | This parameter is the natural frequency (Hz) of the speed control loop. As a guideline, it can be set at up to approximately 1/10 of the natural frequency of the current control loop. However, setting a high-frequency value increases speed ripples due to noise in speed detection. |
| $\zeta_{SG}$ | com_f4_speed_zeta | This parameter is the attenuation coefficient of the speed control loop. It is usually set to 0.7 to 1.0. A value closer to 1 enables a stable and gentle response. |

### 3.7.3 Designing a Position Control Loop

The controller of the position control loop uses only a proportional term. For quick response to an excessive input compared with the speed reference value, the feedforward control for speed is combined to the feedback control to improve responsiveness. The following figure illustrates a model of the position control loop.

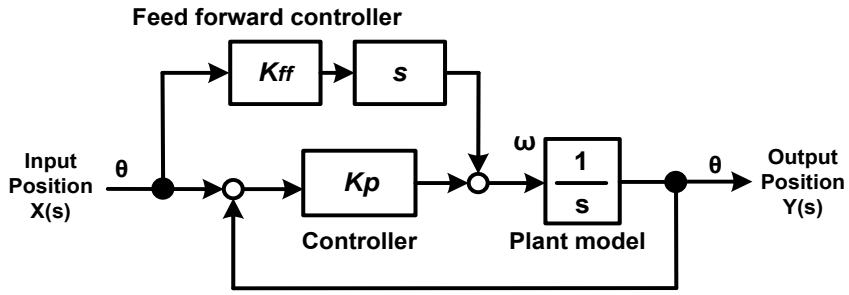



**Figure 3.14 Model of the Position Control Loop**

The position control loop uses only P control and the $K_{p\_position}$ gain is designed using only the natural frequency $\omega_{PG}$ of the position control loop.

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

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

To improve speed responsiveness, the feedforward control for speed is provided.

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

Accordingly, a fixed value is specified as the speed feedforward gain and the position P gain can be designed using the natural frequency $\omega_{PG}$

**Table 3.11 Settings of the Position Control Loop**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $\omega_{PG}$ | com_f4_position_omega | This parameter is the natural frequency (Hz) of the position control loop. As a guideline, it can be set at up to approximately 1/10 of the natural frequency of the speed control loop. |

### 3.7.4 Speed Observer

In the position and speed control using an encoder with a low resolution, speed ripples are generated in the speed information created from the pulse signal due to quantization errors as the speed becomes lower. When the control gain is increased to improve responsiveness, amplified ripples appear as vibration and sound. This vibration will deteriorate the stability of speed control.

An algorithm to estimate speed using a speed observer is provided to reduce speed ripples by the software.

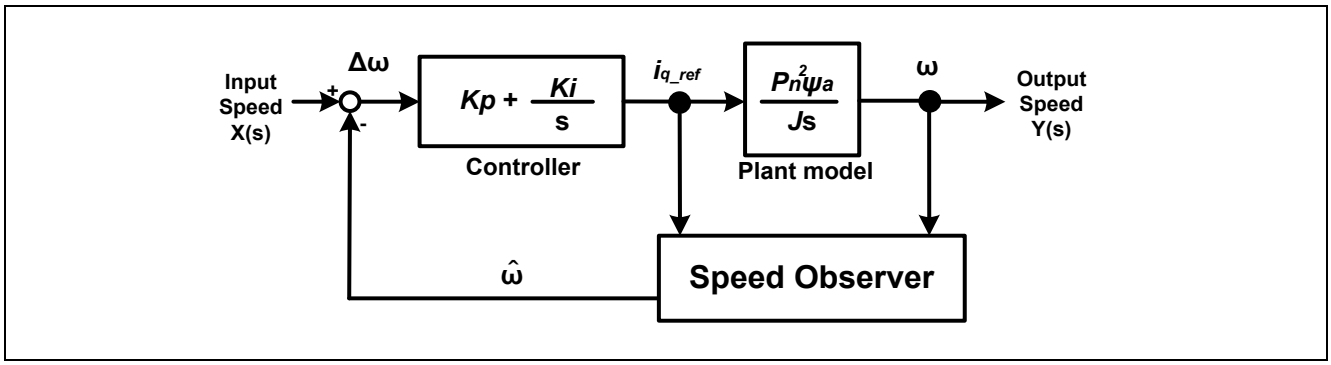



**Figure 3.15 Model of the Speed Control Loop**

The speed observer obtains the estimated speed $\hat{\omega}$ using the q-axis current reference value $i_{q\_ref}$ and speed $\omega$ as inputs. Using the speed observer reduces speed ripples and less affects the responsiveness than normal filter processing.

**Table 3.12 Settings of the Speed Observer**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $\omega_{SOB}$ | com_f4_sob_omega | This parameter is the natural frequency (Hz) of the speed observer. Set a value greater than the natural frequency of the speed control loop. However, setting a high-frequency value increases speed ripples due to noise in speed detection. |
| $\zeta_{SOB}$ | com_f4_sob_zeta | This parameter is the attenuation coefficient of the speed observer. It is usually set to 0.7 to 1.0. A value closer to 1 enables a stable and gentle response. |

### 3.7.5 IPD Controller

If the position resolution or speed resolution is low in the position control loop, a problem of continuous vibrations occurs at the time of positioning due to a failure to respond to slight changes in position difference. To reduce vibrations at the time of positioning, an integral term that accumulates slight changes to eliminate differences is required.

The IPD controller has a control method so that the integral term is only effective for differences and the proportional term and derivative term are only effective for the manipulation amount (controller output). This control method can reduce vibrations at the time of positioning with increased responsiveness.

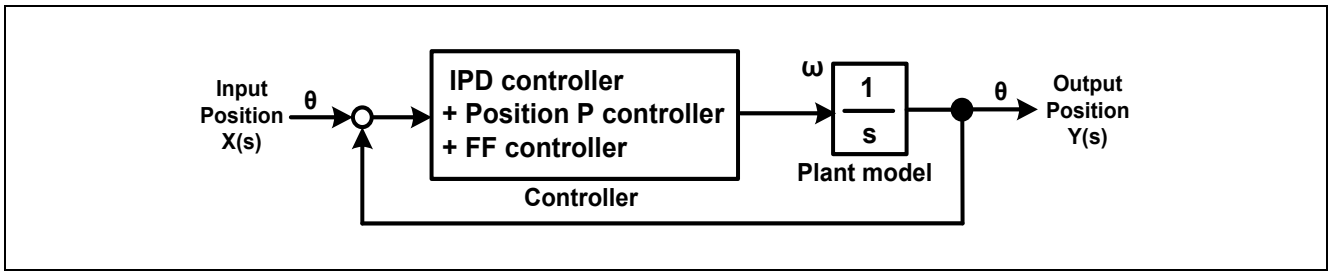



**Figure 3.16 Model of the IPD Controller**

The IPD control implemented in this software is used together with normal proportional control and feedforward control.

### 3.7.6 Voltage Error Compensation

Voltage-type PWM converters have dead time to simultaneously turn off the switching elements of the upper and lower arms to prevent a short circuit between them. For this reason, an error is generated between the voltage reference value and the actual voltage applied to the motor, which deteriorates the control accuracy. To reduce this error, a voltage error compensation function is provided.

The dependence of voltage error on current depends on the direction and intensity of the current, dead time, and switching characteristics of the power elements in use. The dependence on current has the characteristic shown below. The voltage error compensation function compensates a voltage reference value with a voltage pattern opposite to the following voltage error according to the current.

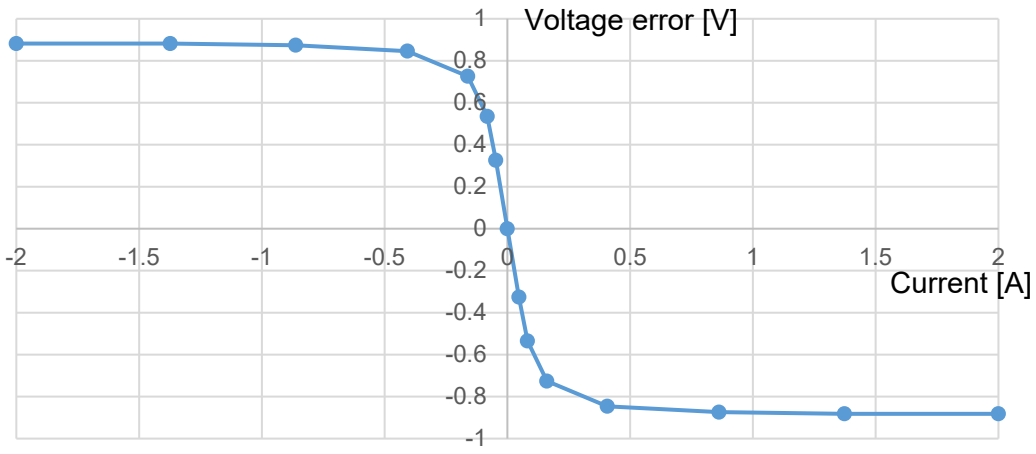



**Figure 3.17 Example of Current Dependence of Voltage Error**

### 3.7.7 Flux-Weakening Control

When the permanent magnet synchronous motor is actuated, an inductive voltage is generated in proportion to the permanent magnetic flux of the rotor and the rotational speed. When the rotational speed increases and the inductive voltage equals the power supply voltage (saturation of voltage), no larger current can flow in the motor and the rotational speed is saturated. That is, there is a trade-off between the torque and speed of rotation of a permanent magnet synchronous motor. For example, the torque of a motor with a powerful magnet is large. However, its inductive voltage is also large, preventing high-speed rotation. The flux-weakening control is provided as a technique to solve this problem.

In flux-weakening control, a d-axis current in the negative direction is applied to suppress the effect of the saturation of voltage due to a voltage induction, achieving high-speed rotation and improving the output over a range of high rotational velocity.

The d-axis current is determined by the following equation.

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

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

$V_{om}$: Inductive voltage limit value (V)

$V_{amax}$: Maximum value of voltage vector (V)

$I_a$: Current vector size (A)

### 3.7.8 Bandpass Filter Delay Compensation

Phase differences are generated at the time of position detection due to the bandpass filters (BPFs) in the RDC.

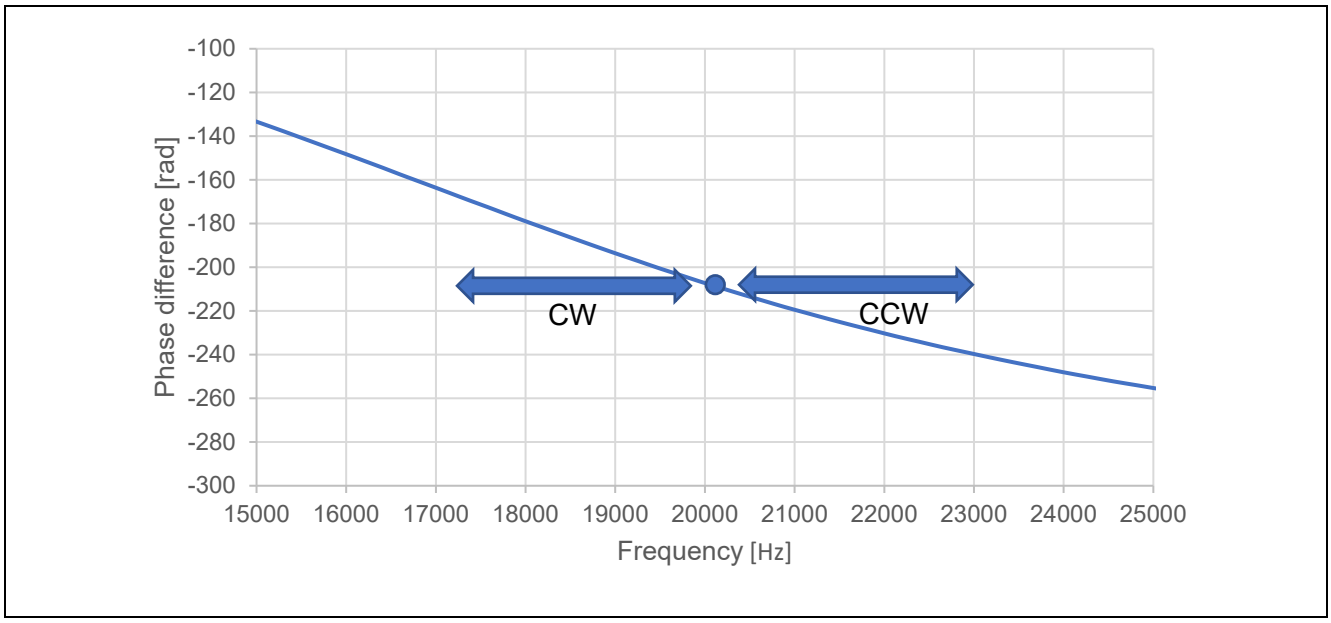



**Figure 3.18 Phase Differences at the Time of Position Detection Due to Bandpass Filters**

As shown in Figure 3.18, the slope of the phase difference with respect to frequency differs between rotation in the clockwise (CW) direction and the counterclockwise (CCW) direction. Therefore, the amount of the compensation is separately determined by linear approximations for the phase difference in each direction of rotation.

The following equation is used to calculate the amount of compensation.

$$\theta_{\mathrm{bpfcomp}} = \begin{cases} \omega C_{base\_cw} C_{mult\_cw} & (\omega \geq 0) \\ \omega C_{base\_ccw} C_{mult\_ccw} & (\omega < 0) \end{cases}$$

$\omega$: Electrical angular speed of motor
$C_{base\_cw}$: BPF base compensation coefficient for the CW direction
$C_{base\_ccw}$: BPF base compensation coefficient for the CCW direction
$C_{mult\_cw}$: BPF compensation coefficient for the CW direction
$C_{mult\_ccw}$: BPF compensation coefficient for the CCW direction

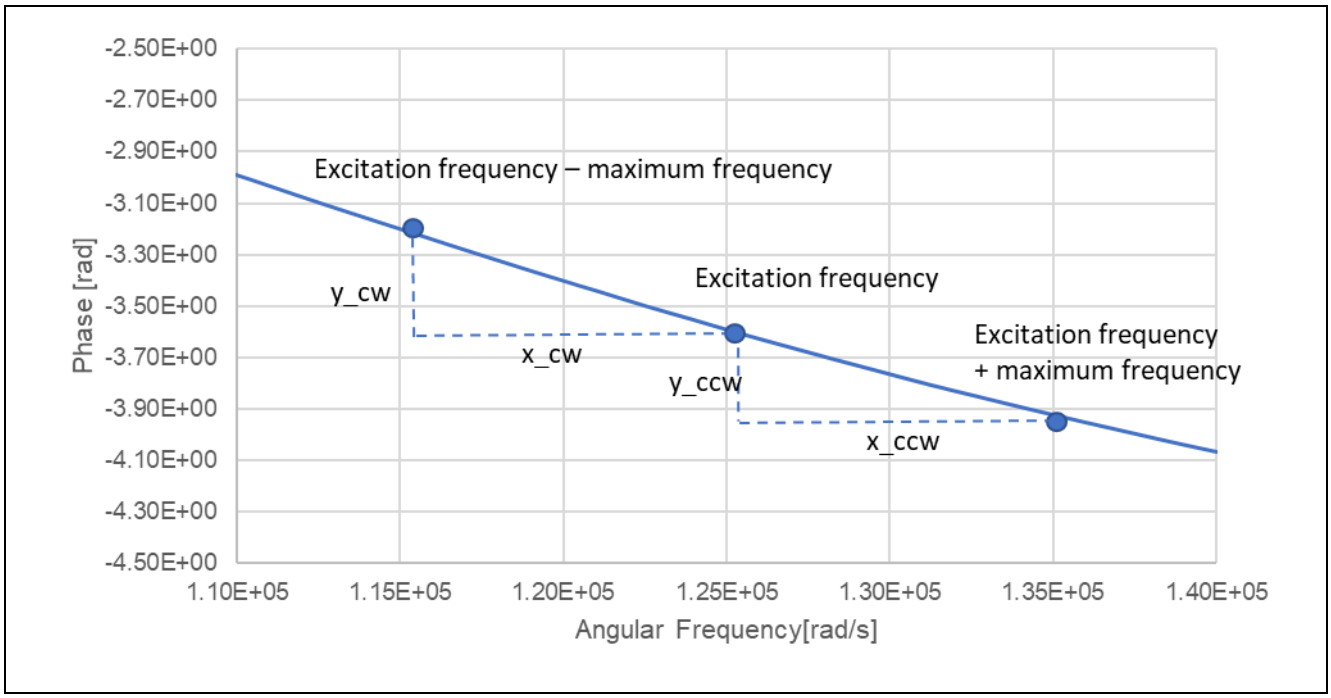



**Figure 3.19 Obtaining BPF Base Compensation Coefficients**

The BPF phase characteristics are determined by the external circuits connected to the RDC. Figure 3.19 shows how to obtain BPF base compensation coefficients.

After the excitation frequency and the maximum operating frequency of the motor are specified, the phase differences y_cw and y_ccw and angular frequency differences x_cw and x_ccw between the two frequencies can be obtained. Use the slopes between the center point and the left and right points (y_cw/x_cw and y_ccw/x_ccw) as compensation coefficients.

$$C_{base\_cw} = \frac{y_{cw}}{x_{cw}}$$

$$C_{base\_ccw} = \frac{y_{ccw}}{x_{ccw}}$$

Specify the obtained base compensation coefficients in the RMW variables shown in Table 3.13.

**Table 3.13 Settings of BPF Compensation**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $C_{base\_cw}$ | com_f4_bpf_comp_base_cw | BPF base compensation coefficient for the clockwise direction<br>This parameter is set according to the characteristics of the external BPF circuit for the RDC. |
| $C_{base\_ccw}$ | com_f4_bpf_comp_base_ccw | BPF base compensation coefficient for the counterclockwise direction<br>This parameter is set according to the characteristics of the external BPF circuit for the RDC. |
| $C_{mult\_cw}$ | — | BPF compensation coefficient for the clockwise direction<br>Always fixed to 1. |
| $C_{mult\_ccw}$ | — | BPF compensation coefficient for the counterclockwise direction<br>Always fixed to 1. |

### 3.7.9 Iron Loss Compensation

As shown in the equivalent circuit of a motor (Figure 3.20), a motor has an iron loss due to the iron loss resistance represented by Ri as an internal magnetic loss in addition to a copper loss due to the coil resistance represented by R. Iron loss will particularly affect a stepping motor in operation in the high-velocity range.

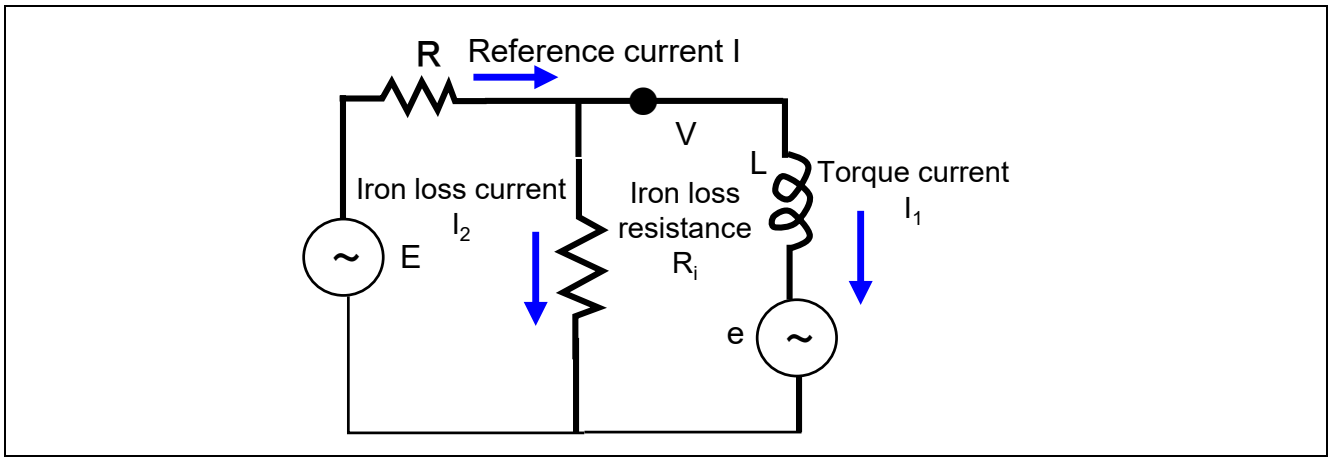



**Figure 3.20 Iron Loss in the Equivalent Circuit of a Motor**

As the iron loss current $I_2$ flows due to the iron loss in the high-velocity range, the phase and size of the torque current $I_1$, which affects the torque, varies with respect to the torque reference current I as shown in Figure 3.21 and the phase of the torque current $I_1$, which should be aligned with the q axis, is delayed with respect to that of the reference current I. Therefore, obtaining the desired torque will not be possible and the performance in control will be degraded unless the torque current includes compensation for the phase delay due to the iron loss.

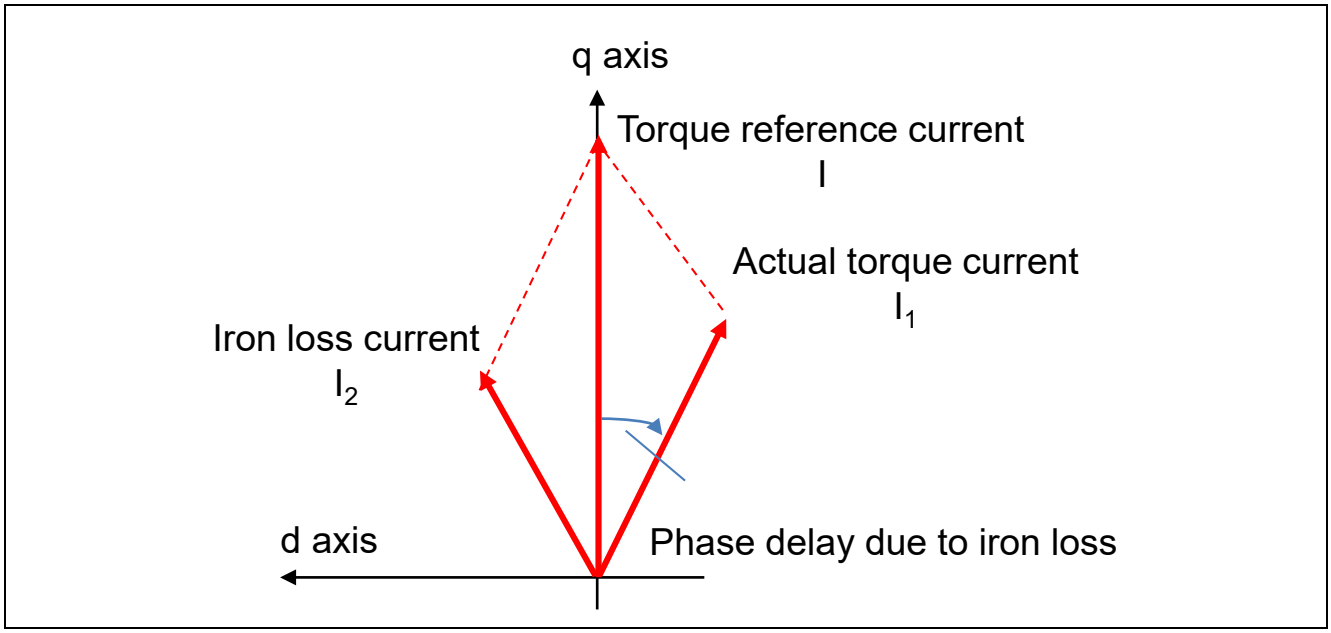



**Figure 3.21 Phase Delay Due to Iron Loss**

This control software only compensates for the phase delay due to the iron loss by using the following equation.

$$\theta_{\mathrm{comp}} = \omega C_{base}$$

**Table 3.14 Setting of Iron Loss Compensation**

| Parameter Symbol | RMW Variable | Setting Method |
|---|---|---|
| $C_{base}$ | com_f4_iron_loss_comp_base | Compensation coefficient for iron loss delay<br>This value is set by tuning according to the motor type.<br>It is set to $0.65 \times 10^{-5}$ by default. |

If the motor's rotational speed does not reach the theoretical maximum value, repeat the adjustment of this compensation parameter in 5% steps (as a rough suggested value) while monitoring the operation of the motor because actually measuring the iron loss itself is difficult.

## 3.8 System Protection Functions

This control program has functions for detecting the following error states and making an emergency stop. For the settings of each system protection function, see Table 3.15.

- Overcurrent error
  An overcurrent error is detected by hardware and software.
  Overcurrent detection by hardware:
  The emergency stop signal (overcurrent detection) from the hardware circuit drives the PWM output pins to high impedance.
  Overcurrent detection by software:
  If an overcurrent (exceeding the overcurrent limit value) is detected during monitoring of the phase-α and phase-β currents at overcurrent monitoring intervals, emergency stop is activated.
  The overcurrent limit value is automatically calculated from the rated current MP_NOMINAL_CURRENT_RMS of the motor.
- Overvoltage error
  If an overvoltage (exceeding the overvoltage limit value) is detected during monitoring of the inverter bus voltage at overvoltage monitoring intervals, emergency stop is activated. The overvoltage limit value is set considering the error in the resistance value of the detection circuit.
- Low-voltage error
  If a low voltage (below the low-voltage limit value) is detected during monitoring of the inverter bus voltage at low-voltage monitoring intervals, emergency stop is activated. The low-voltage limit value is set considering the error in the resistance value of the detection circuit.
- Rotation speed error
  If a speed limit value is exceeded during monitoring of the speed at rotational speed monitoring intervals, emergency stop is activated.
- Disconnection or overheat detection error (RDC alarm detection signal)
  If a low-level signal due to overheat of the RDC is detected, emergency stop is activated.

**Table 3.15 Settings of Each System Protection Function**

| | | |
|---|---|---|
| Overcurrent error | Overcurrent limit value (A) | 3 |
| | Monitoring interval (μs) | 50 |
| Overvoltage error | Overvoltage limit value (V) | 40 |
| | Monitoring interval (μs) | 50 |
| Low-voltage error | Low-voltage limit value (V) | 18 |
| | Monitoring interval (μs) | 50 |
| Rotational speed error | Speed limit value (rpm) | 4000 |
| | Monitoring interval (μs) | 50 |

## 3.9 Calibration to Compensate for Errors

This program has functions to implement automatic calibration through calls of the RDC driver functions that run the automatic calibration facilities of the RDC. For details on automatic calibration by the RDC driver, see the application note Using the Driver for Resolver-to-Digital Converter Control (R03AN0013). Pressing push switch 2 (SW2) enables the above automatic calibration to compensate for errors and the elimination of offsets from the actual angle. On the latter point, see section 3.5.

## 3.10 Specifications of Software Functions

The control processing in this software mainly consists of 50-µs cycle interrupt (carrier interrupt) processing, 250-µs cycle interrupt processing, and external interrupt (error handling and reference pulse input) processing. In the figure below, the yellow region shows 50-µs cycle interrupt processing and the blue region shows 250-µs cycle interrupt processing.

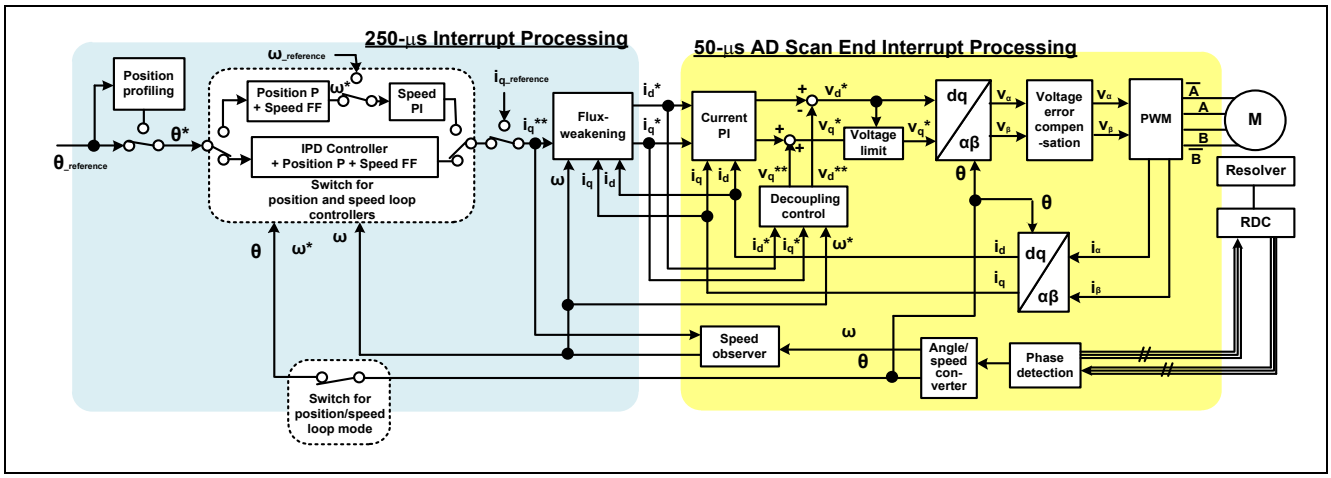



**Figure 3.22 Block Diagram of the Vector Control System**

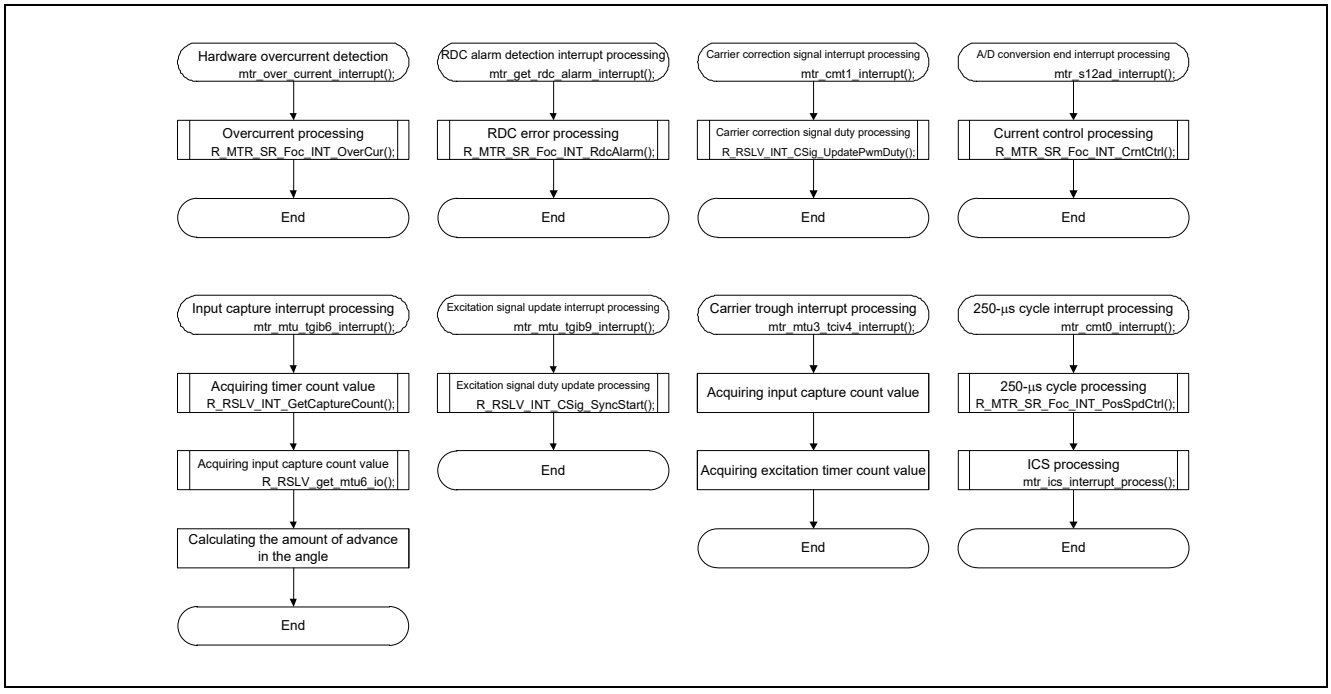



**Figure 3.23 Interrupt Processing**

**Table 3.16 Interrupt Functions**

| File Name | Function | Processing | Interval |
|---|---|---|---|
| driver/mcu/ r_mtr_interrupt.c | mtr_get_rdc_alarm_interrupt<br>Input: None<br>Output: None | RDC error processing<br>• Calls the error processing function mtr_rdc_alarm(). | — |
| | mtr_over_current_interrupt<br>Input: None<br>Output: None | Hardware overcurrent detection (POE) interrupt<br>• Calls the overcurrent processing function mtr_over_current(). | — |
| | mtr_cmt0_interrupt<br>Input: None<br>Output: None | 250-µs cycle processing | 4 kHz |
| | mtr_cmt1_interrupt<br>Input: None<br>Output: None | Updates the carrier correction signal. | 40 kHz |
| | rslv_capture_interrupt<br>Input: None<br>Output: None | Acquires the angle at the time of input capture. | — |
| | rslv_esig_interrupt<br>Input: None<br>Output: None | Updates the resolver excitation signal. | 40 kHz |
| | mtr_mtu3_tciv4_interrupt<br>Input: None<br>Output: None | Acquires the angle at the time of current detection. | 20 kHz |
| | mtr_s12ad_interrupt<br>Input: None<br>Output: None | This is called each time A/D conversion finishes.<br>• Calls the current control processing function mtr_foc_interrupt_carrier(). | 20 kHz |

### 3.10.1 Current Control Processing (R_MTR_SR_Foc_INT_CrntCtrl())

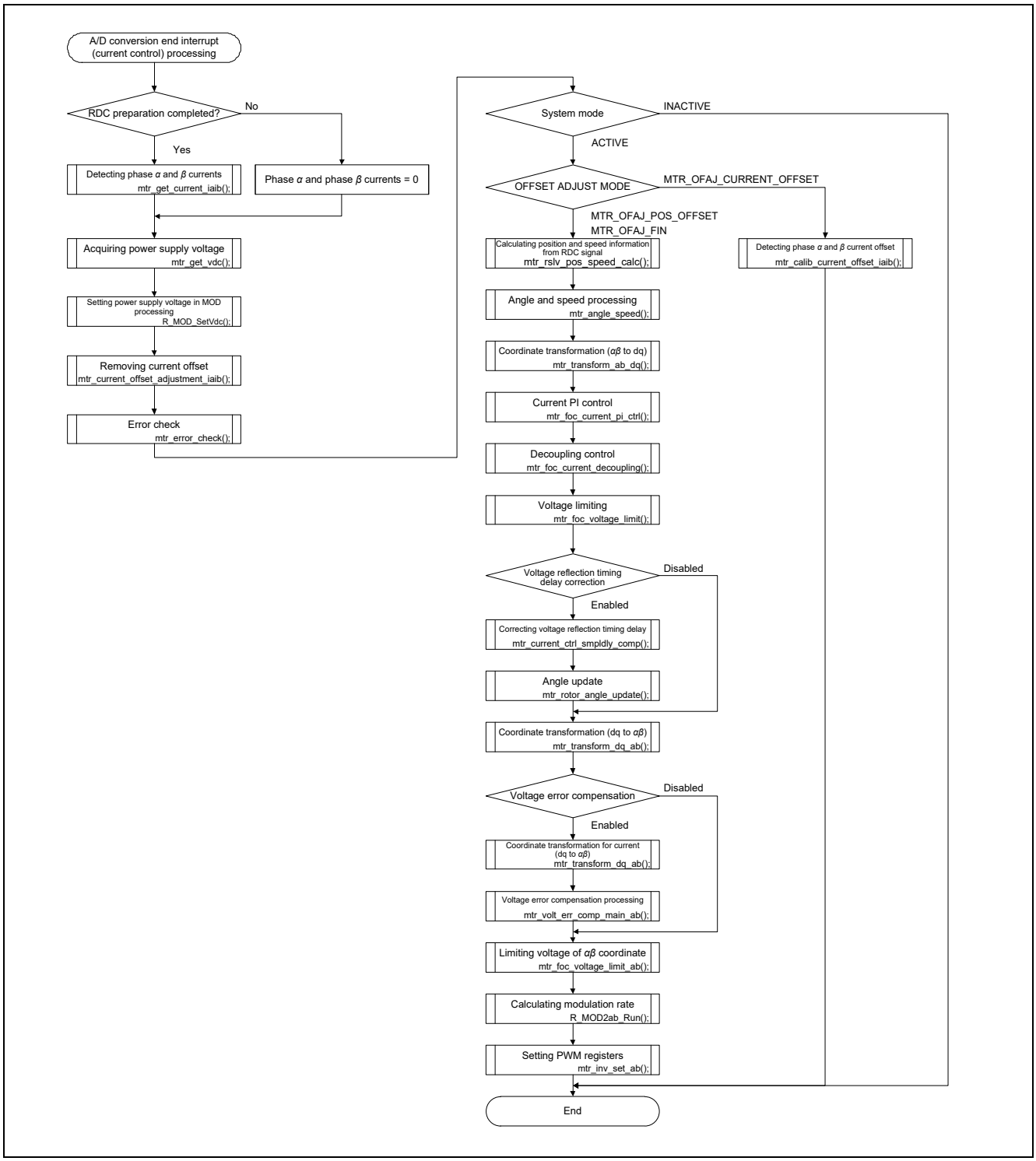



**Figure 3.24 Current Control Processing Flow**

**Table 3.17 Current Control Processing Functions (1/3)**

| File Name | Function | Processing |
|---|---|---|
| /driver/inverter/<br>r_mtr_ctrl_inverter.c | mtr_get_current_iaib<br>Input: float *f4_ia_ad:<br>Pointer to phase α current variable<br>float *f4_ib_ad:<br>Pointer to phase β current variable<br>uint8_t u1_id:<br>Motor ID<br>Output: None | Acquires the αβ-axis currents. |
| | mtr_get_vdc<br>Input: uint8_t u1_id: Motor ID<br>Output: None | Acquires the power-supply voltage value. |
| /r_mtr_spm_rslv_foc_rx/src/functions/<br>r_mtr_mod.c | mtr_mod_SetVdc<br>Input: mod_t *p_mode:<br>Pointer to modulation structure<br>float f4_vdc: Power-supply voltage<br>Output: None | Passes the power-supply voltage value to the modulation module. |
| /r_mtr_spm_rslv_foc_rx/src/controller/<br>r_mtr_foc_stm_rslv_control.c | mtr_current_offset_adjustment_iaib<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Phase α and phase β current offset detection |
| | mtr_error_check<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Checks errors. |
| /r_mtr_spm_rslv_foc_rx/src/mode/<br>r_mtr_statemachine.c | mtr_statemachine_get_status<br>Input:<br>mtr_statemachine_t *p_state_machine:<br>Pointer to state machine structure<br>Output: uint8_t:<br>State transition information<br>MTR_MODE_INACTIVE<br>MTR_MODE_ACTIVE<br>MTR_MODE_ERROR | Acquires the state transition information. |
| /r_mtr_spm_rslv_foc_rx/src/controller/<br>r_mtr_foc_stm_rslv_control.c | mtr_calib_current_offset_iaib<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Phase α and phase β current offset detection |
| | mtr_rslv_pos_speed_calc<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Calculates angle and speed values from the resolver signal. |
| | mtr_angle_speed<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Reflects the angle and speed in the control system. |
| /r_mtr_spm_rslv_foc_rx/src/controller/<br>r_mtr_transform.c | mtr_transform_ab_dq<br>Input: mtr_rotor_angle_t *p_angle:<br>Pointer to angle management structure<br>float *f4_ab: Pointer to αβ-axis variable<br>float *f4_dq: Pointer to dq-axis variable<br>Output: None | Convers coordinates from αβ-axis to dq-axis system. |

**Table 3.17 Current Control Processing Functions (2/3)**

| File Name | Function | Processing |
|---|---|---|
| /r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_current.c | mtr_foc_current_pi_ctrl<br>Input: mtr_current_control_t *st_cc:<br>Pointer to current control structure<br>Output: None | Executes current PI control. |
| | mtr_foc_current_decoupling<br>Input: mtr_current_control_t *st_cc:<br>Pointer to current control structure<br>float f4_speed_rad: Speed (rad/s)<br>tr_parameter_t *p_mtr:<br>Pointer to motor parameter structure<br>Output: None | Executes decoupling control processing. |
| /r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_stm_rslv_control.c | mtr_foc_voltage_limit<br>Input: mtr_foc_control_t *p_foc:<br>Pointer to vector control structure<br>Output: None | Limits dq-axis voltages. |
| | mtr_current_ctrl_smpldly_comp<br>Input: float f4_angle_rad: Angle (rad)<br>float f4_speed_rad: Speed (rad/s)<br>Output: float: Angle (rad) | Calculates the correction value for the angle delay from current detection to voltage reflection. |
| /r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_transform.c | mtr_rotor_angle_update<br>Input: mtr_rotor_angle_t *p_angle:<br>Pointer to angle management structure<br>float f4_angle_rad: Angle (rad)<br>Output: None | Updates the coordinate transformation angle. |
| | mtr_transform_dq_ab<br>Input: mtr_rotor_angle_t *p_angle:<br>Pointer to angle management structure<br>float *f4_dq: Pointer to dq- axis variable<br>float *f4_ab: Pointer to αβ-axis variable<br>Output: None | Converts coordinates from the αβ-axis system to the dq-axis system. |
| /r_mtr_spm_rslv_foc_rx/src /functions/ r_mtr_volt_err_comp.obj | mtr_volt_err_comp_main_ab<br>Input: mtr_volt_comp_t *st_volt_comp:<br>Pointer to voltage error compensation structure<br>float *p_f4_v_array:<br>Pointer to αβ voltage<br>float *p_f4_i_array: Pointer to αβ current<br>float f4_vdc: Power-supply voltage<br>Output: None | Calculates the voltage error. |
| /r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_stm_rslv_control.c | mtr_foc_voltage_limit_ab<br>Input: float *f4_v_ab: Pointer to αβ voltage<br>float f4_vdc: Power-supply voltage<br>Output: None | Limits the phase α and phase β voltage. |

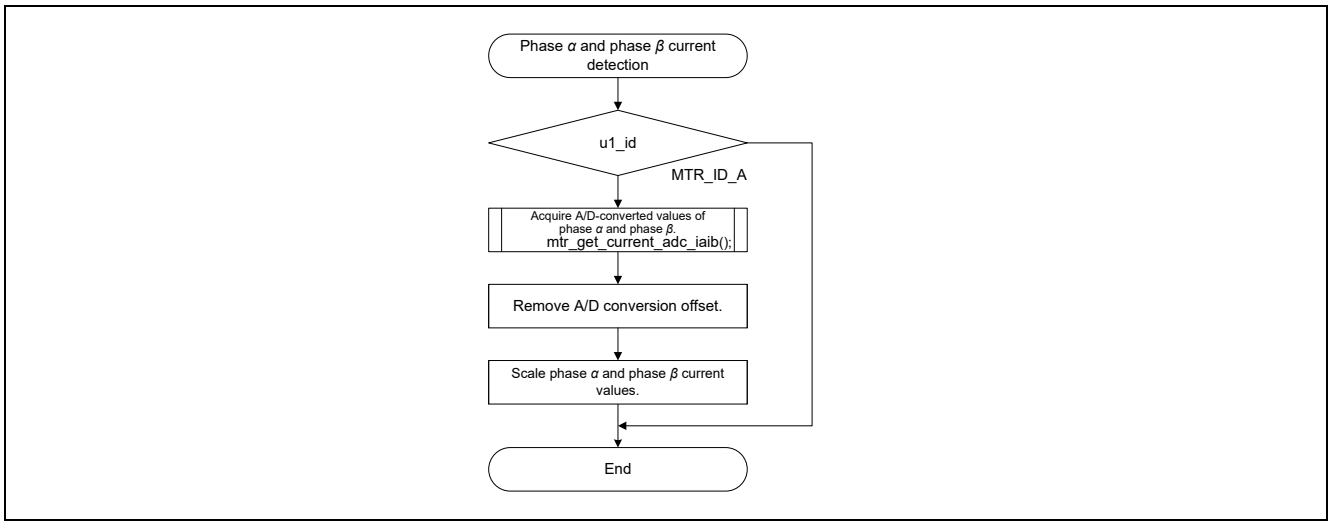
**Table 3.17 Current Control Processing Functions (3/3)**

| File Name | Function | Processing |
|---|---|---|
| /r_mtr_spm_rslv_foc_rx/src<br>/functions/<br>r_mtr_mod.c | mtr_mod2ab_run<br>Input: float f4_va: α-axis voltage<br>float f4_vb: β-axis voltage<br>float f4_vdc: Power-supply voltage<br>float *f4_moda: α-axis modulation rate<br>float *f4_modb: β-axis modulation rate<br>Output: None | Calculates the modulation rate. |
| /driver/mcu/<br>r_mtr_ctrl_rx24t.c | mtr_inv_set_ab<br>Input: float f4_duty_a: α-axis modulation rate<br>float f4_duty_b: β-axis modulation rate<br>uint8_t u1_id: Motor ID<br>Output: None | Sets information in PWM registers. |

## mtr_get_current_iaib

Phase α and phase β current acquisition

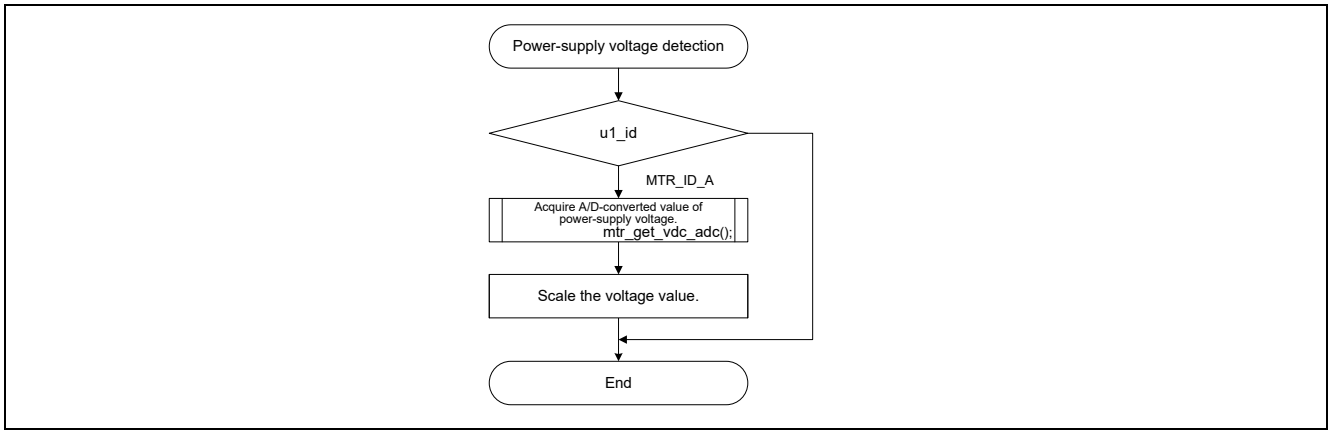
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_ctrl_inverter.h"<br>void mtr_get_current_ iaib ( float *f4_ia_ad, float *f4_ib_ad, uint8_t u1_id ); | | |
| Argument | float *f4_ia_ad | I | Pointer to the phase α current variable |
| | float *f4_ib_ad | I | Pointer to the phase β current variable |
| | uint8_t | I | Motor ID |
| Return value | None | | |
| Description | Converts the values detected by the A/D converter to current values, and then passes the current values to the phase α and phase β current value variables.<br>This function acquires the current values of the motor specified by the motor ID variable. | | |
| Note | When this function is called, the function that returns the A/D-converted values is called, and then offset removal and scaling are automatically handled. | | |



## mtr_get_vdc

Power-supply voltage acquisition

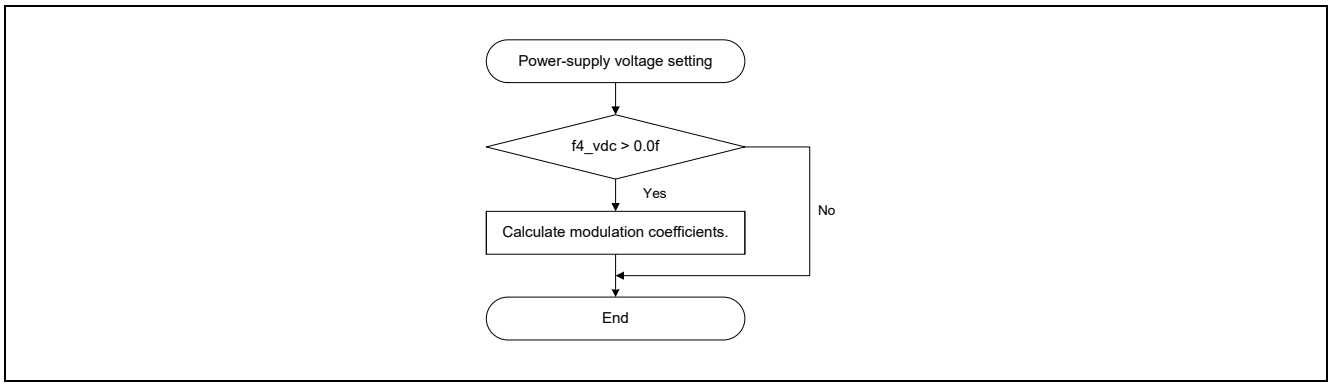
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_ctrl_inverter.h"<br>void mtr_get_vdc ( uint8_t u1_id ); | | |
| Argument | uint8_t | I | Motor ID |
| Return value | None | | |
| Description | Passes the value detected by the A/D converter to the power supply voltage value variable.<br>This function acquires the voltage value of the motor specified by the motor ID variable. | | |
| Note | | | |



## mtr_mod_SetVdc

Passing the power-supply voltage value to the modulation module

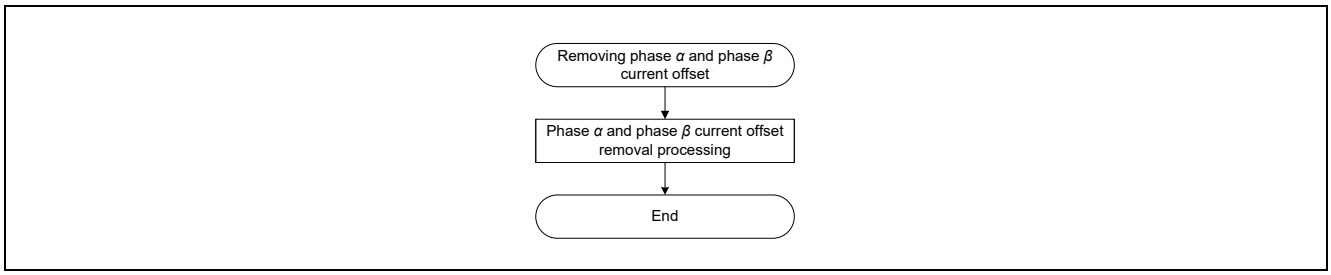
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_mod.h"<br>void mtr_mod_SetVdc ( mod_t *p_mod, float f4_vdc ); | | |
| Argument | mod_t *p_mod | I | Pointer to the modulation structure |
| | float f4_vdc | I | Power-supply voltage |
| Return value | None | | |
| Description | Specifies the power-supply voltage in the modulation module. | | |
| Note | Specification of the value of 0 or a negative power-supply voltage is ignored because such a value cannot be specified as the power-supply voltage. | | |



## mtr_get_current_offset_adjustment_iaib

Removing phase α and phase β current offset

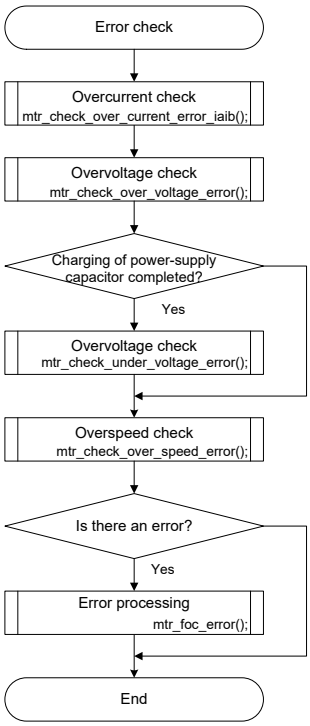
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>void mtr_current_offset_adjustment_iaib ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I | Pointer to the vector control structure |
| Return value | None | | |
| Description | Removes the detected current offset value from the phase α and phase β current values. | | |
| Note | Use this function together with the current offset detection function. | | |



## mtr_error_check

Error check

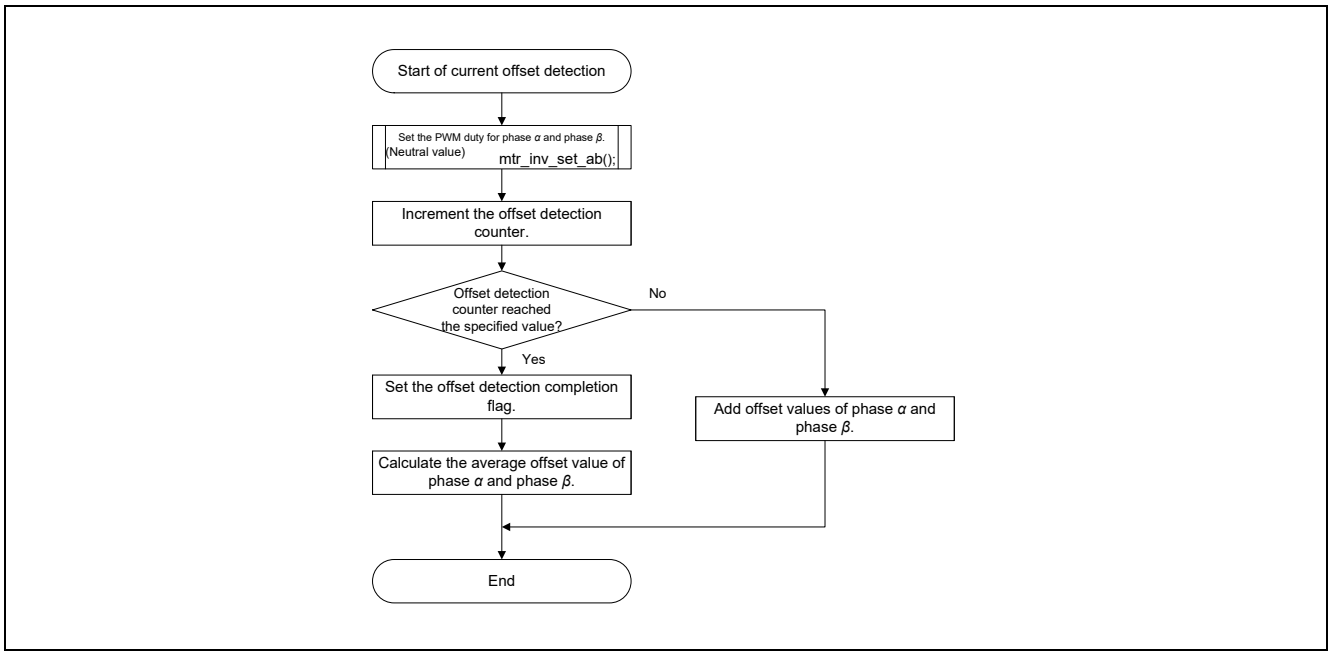
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>void mtr_error_check ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I | Pointer to the vector control structure |
| Return value | None | | |
| Description | Checks the current, power-supply voltage, and speed to detect overcurrent, overvoltage, low-speed, and overspeed errors. | | |
| Note | | | |



## mtr_calib_current_offset_iaib

Phase α and phase β current offset detection processing

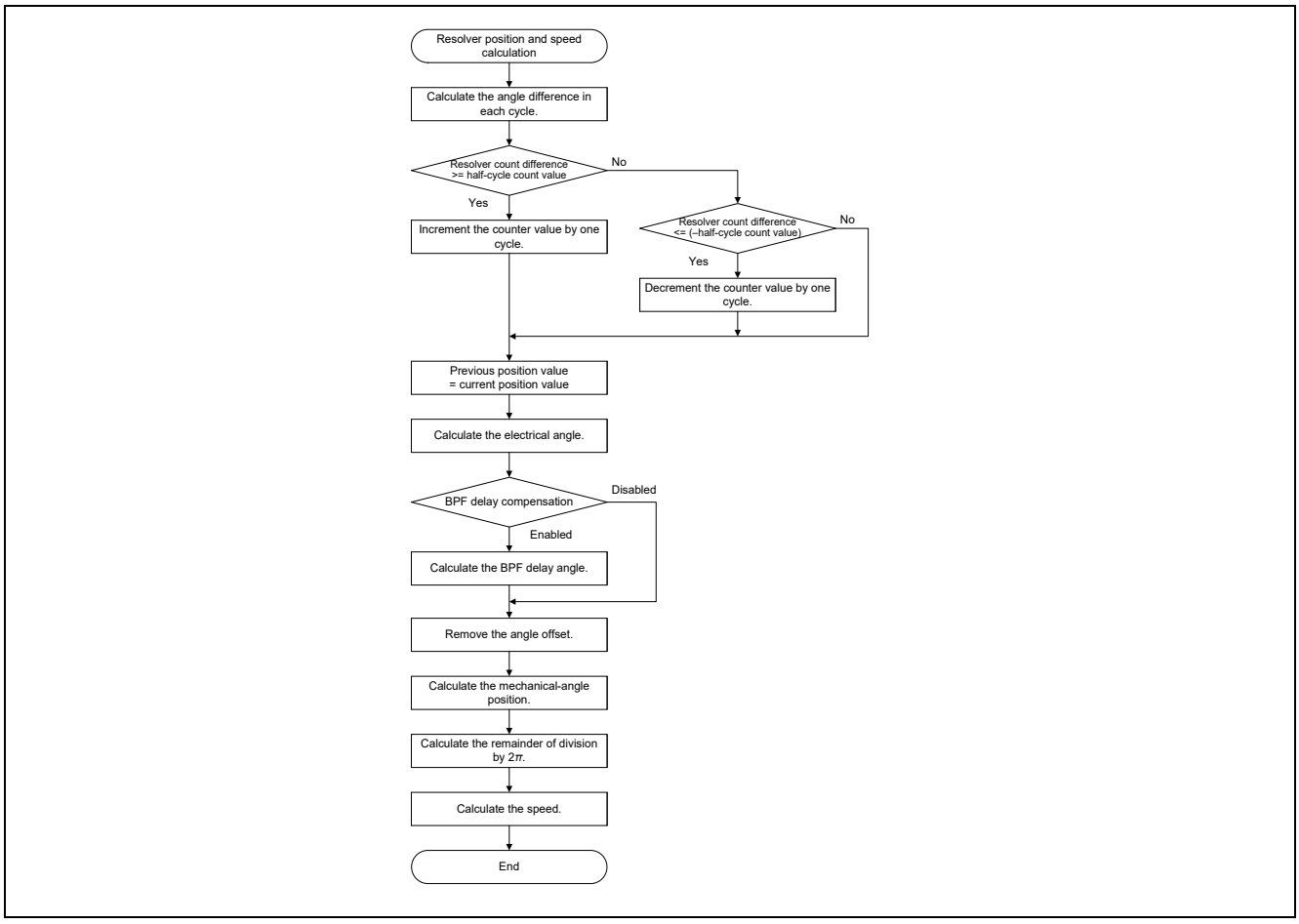
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>void mtr_calib_current_offset_iaib ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | None | | |
| Description | Detects the current offset when the output current is zero. | | |
| Note | | | |



## mtr_rslv_pos_speed_calc

Resolver position and speed calculation

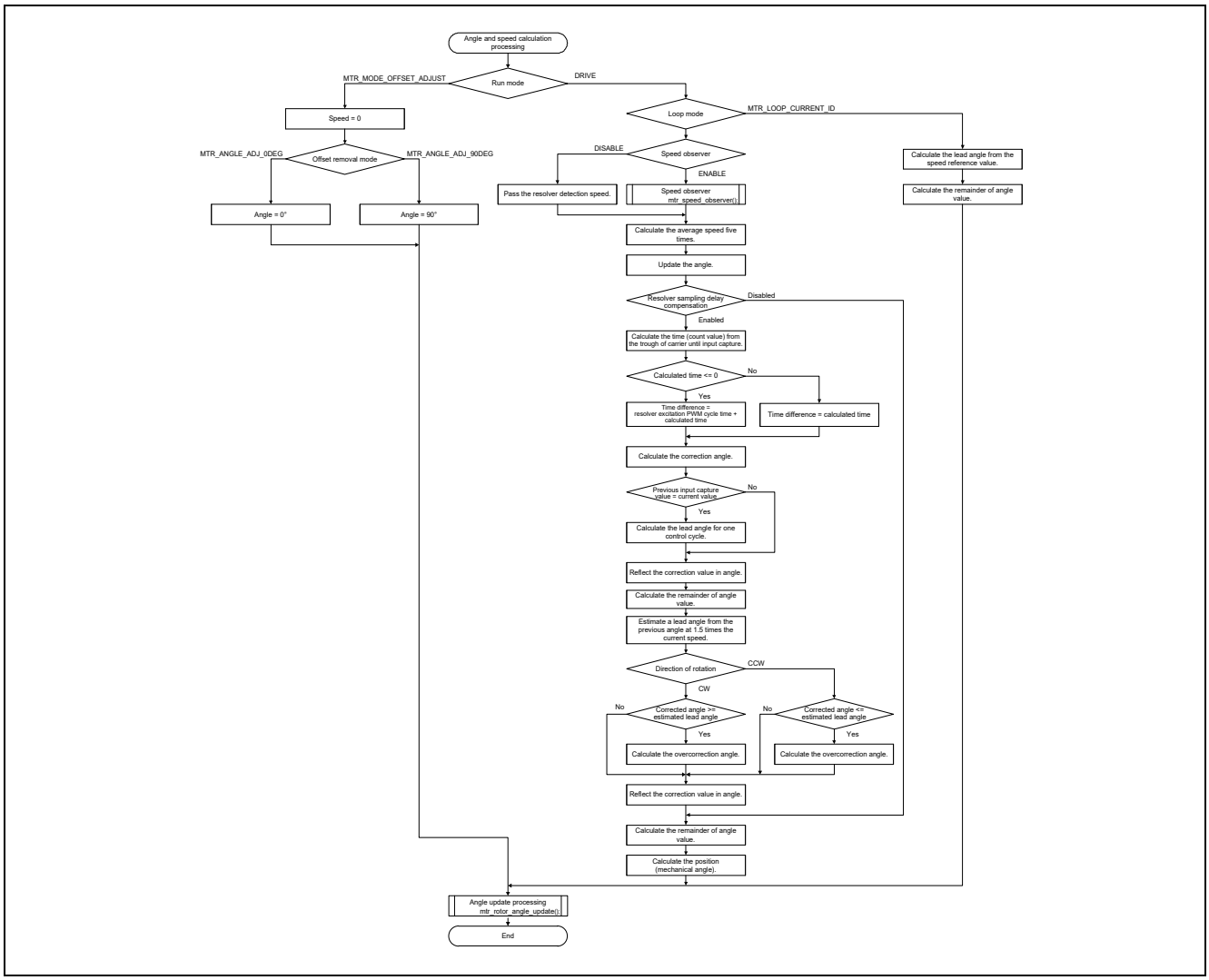
| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_foc_stm_rslv_position.h "<br>void mtr_rslv_pos_speed_calc ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | None | | |
| Description | Calculates the position and speed from the angle information sent from the RDC.<br>Independently manages the mechanical-angle position and the position detected by the resolver sensor.<br>Calculates the speed from the mechanical-angle position. | | |
| Note | | | |



## mtr_angle_speed

Angle and speed calculation processing

| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_foc_stm_rslv_position.h "<br>void mtr_angle_speed ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | None | | |
| Description | Calculates the angle and speed in each drive mode. | | |
| Note | | | |



## mtr_transform_ab_dq

αβ-to-dq coordinate system conversion

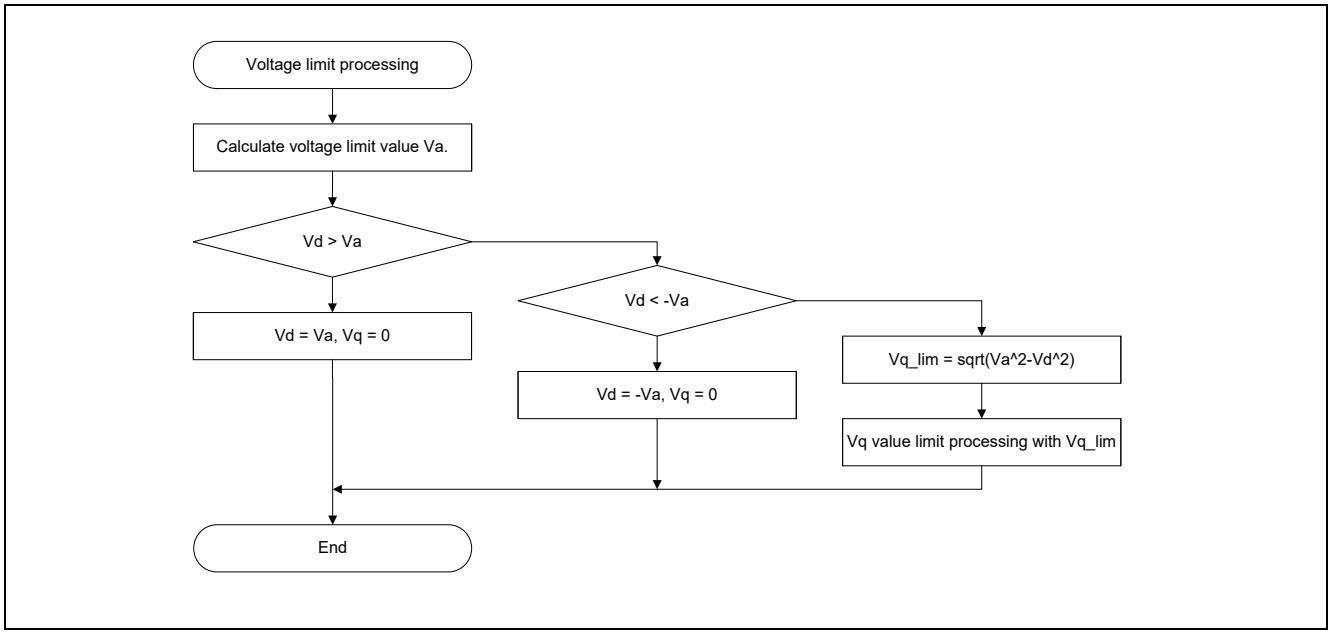
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_transform.h"<br>void mtr_transform_ab_dq (const mtr_rotor_angle_t *p_angle,<br>const float *f4_ab, float *f4_dq ); | | |
| Argument | const mtr_rotor_angle_t *p_angle | I | Pointer to the rotor angle structure |
| | const float *f4_ab | I | Pointer to the αβ-axis variable |
| | float *f4_dq | O | Pointer to the dq-axis variable |
| Return value | None | | |
| Description | Converts the αβ coordinate system to the dq coordinate system. | | |
| Note | The αβ-axis and dq-axis variables assume that the addresses of α axis and β axis are consecutive and the addresses of d axis and q axis are consecutive. The start address of each variable is referenced. | | |

$$\begin{bmatrix} d \\ q \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

## mtr_foc_voltage_limit

Voltage limit processing

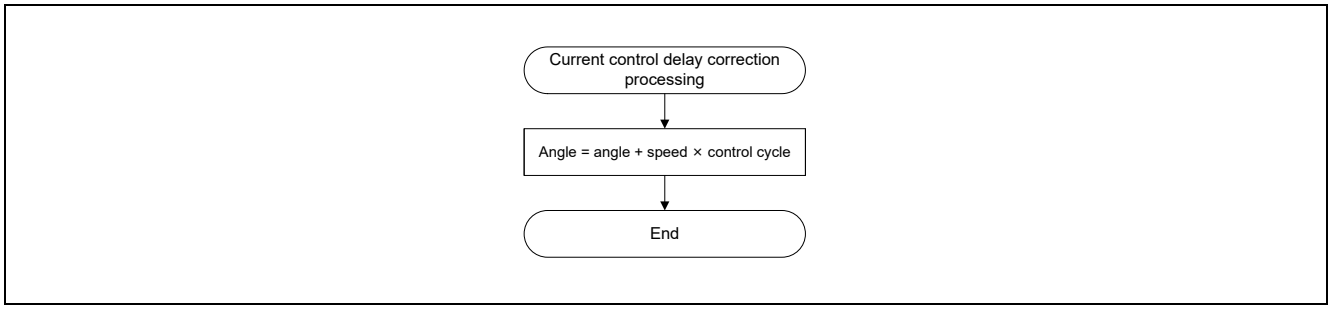
| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_foc_stm_rslv_control.h"<br>void mtr_foc_voltage_limit ( mtr_foc_control_t *p_foc ); | | |
| Argument | mtr_foc_control_t *p_foc | I/O | Pointer to the vector control structure |
| Return value | None | | |
| Description | Handles limit processing for the dq-axis voltage reference value. | | |



## mtr_current_ctrl_smpldly_comp

Current control delay correction

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>float mtr_current_ctrl_smpldly_comp (float f4_angle_rad, float f4_speed_rad); | | |
| Argument | float f4_angle_rad | I | Angle (rad) |
| | float f4_speed_rad | I | Speed (rad/s) |
| Return value | float | Angle (rad) | |
| Description | Corrects the delay of angle. | | |
| Note | | | |



## mtr_transform_dq_ab

dq-to-αβ coordinate system conversion

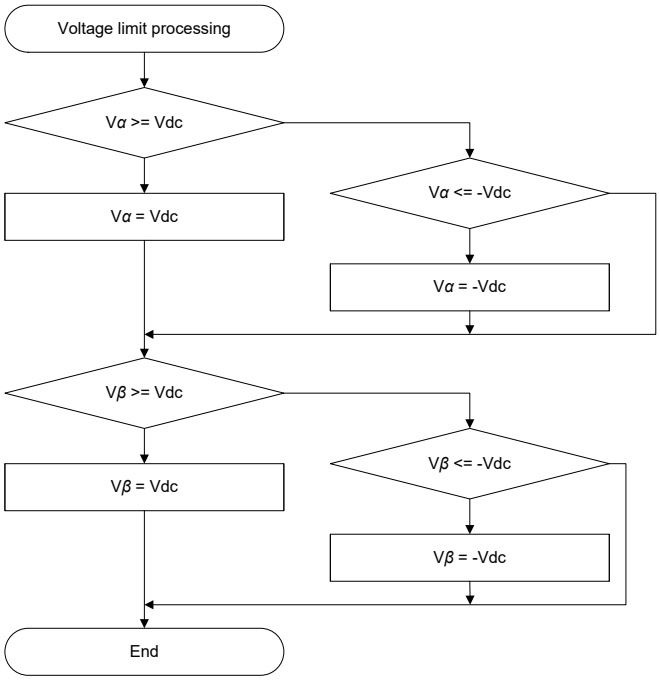
| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_transform.h"<br>void mtr_transform_dq_ab (const mtr_rotor_angle_t *p_angle,<br>const float *f4_dq, float *f4_ab,); | | |
| Argument | const mtr_rotor_angle_t *p_angle | I | Pointer to the rotor angle structure |
| | const float *f4_dq | I | Pointer to the dq-axis variable |
| | float *f4_ab | O | Pointer to the αβ-axis variable |
| Return value | None | | |
| Description | Converts the dq coordinate system to the αβ coordinate system. | | |
| Note | The αβ-axis and dq-axis variables assume that the addresses of α axis and β axis are consecutive and the addresses of d axis and q axis are consecutive. The start address of each variable is referenced. | | |

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}cos\theta & -sin\theta\\ sin\theta & cos\theta\end{bmatrix}\begin{bmatrix}d\\ q\end{bmatrix}$$

## mtr_foc_voltage_limit_ab

αβ-axis voltage limit processing

| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_foc_stm_rslv_control.h"<br>void mtr_foc_voltage_limit_ab (float *f4_v_ab, float f4_vdc) | | |
| Argument | float *f4_v_ab | I/O | Pointer to the αβ-axis voltage |
| | float f4_vdc | I | Power-supply voltage |
| Return value | None | | |
| Description | Handles limit processing for the αβ-axis voltage reference value. | | |
| Note | | | |



## mtr_mod2ab_run

Duty value calculation processing

| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_mod.h"<br>void mtr_mod2ab_run ( float f4_va, float f4_vb, float f4_vdc,<br>float *f4_moda, float *f4_modb ); | | |
| Argument | float f4_va | I | α-axis voltage |
| | float f4_vb | I | β-axis voltage |
| | float f4_vdc | I | Power-supply voltage |
| | float *f4_moda | O | α-axis duty value |
| | float *f4_modb | O | β-axis duty value |
| Return value | None | | |
| Description | Calculates duty values of the α axis and β axis.<br>Duty values are converted so that the range of –Vdc to Vdc can be expressed with a duty value of 0 to 1. | | |
| Note | This function processing varies with board specifications. | | |

$$mod_a = \frac{V_a}{V_{dc}} * 0.5 + 0.5$$

$$mod_b = \frac{V_b}{V_{dc}} * 0.5 + 0.5$$

## mtr_inv_set_ab

PWM register setting

| | | | |
|---|---|---|---|
| Syntax | #include " r_mtr_ctrl_rx24t.h"<br>void mtr_inv_set_ab ( float f4_moda, float f4_modb, uint8_t u1_id ); | | |
| Argument | float f4_moda | I | α-axis duty value |
| | float f4_modb | I | β-axis duty value |
| | uint8_t u1_id | I | Motor ID |
| Return value | None | | |
| Description | Sets the αβ-axis duty values.<br>This function sets the duty values for the motor specified by the motor ID. | | |
| Note | | | |

### 3.10.2 250-µs Cycle Interrupt Processing (R_MTR_SR_Foc_INT_PosSpdCtrl())

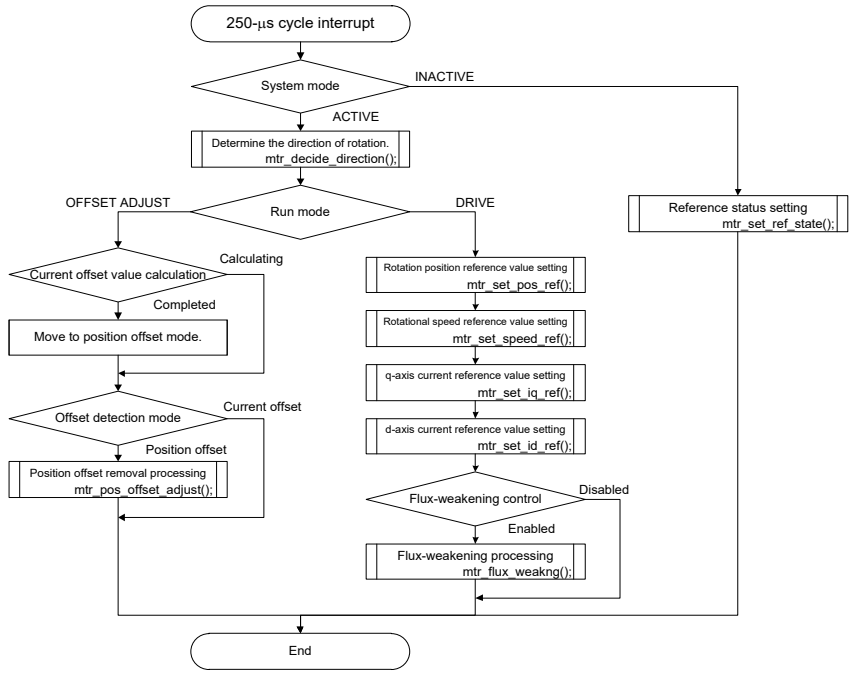



**Figure 3.25 250-µs Cycle Interrupt Processing Flow**

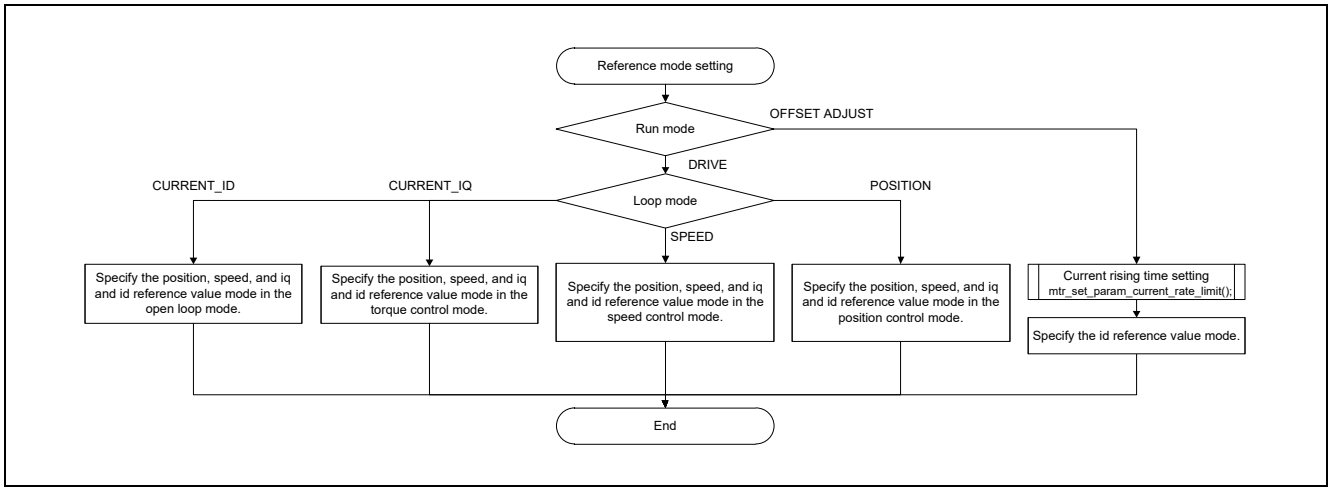
**Table 3.18 250-μs Cycle Interrupt Processing Functions**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_stm_rslv_control.c | mtr_set_ref_state<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: None | Sets reference status of each control loop. |
| | mtr_pos_offset_adjust<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: float: d-axis current reference value (A) | Removes the position offset. |
| | mtr_set_id_ref<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: float: d-axis current reference value (A) | Creates an id reference value. |
| | mtr_set_iq_ref<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: float: q-axis current reference value [A] | Creates an iq reference value. |
| | mtr_set_speed_ref<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: float: Speed reference value (rad/s) | Creates a speed reference value. |
| | mtr_set_pos_ref<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>Output: float: Position reference value (rad) | Creates a position reference value. |
| | mtr_flux_weakng<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>float *f4_idq_ref: Pointer to dq-axis current<br>Output: None | Executes flux-weakening control. |

## mtr_set_ref_state

Reference mode setting of each control loop

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>void mtr_set_ref_state (mtr_foc_control_t *st_foc); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | None | | |
| Description | Sets the reference mode for position, speed, and dq-axis current reference values. | | |
| Note | | | |



## mtr_pos_offset_adjust

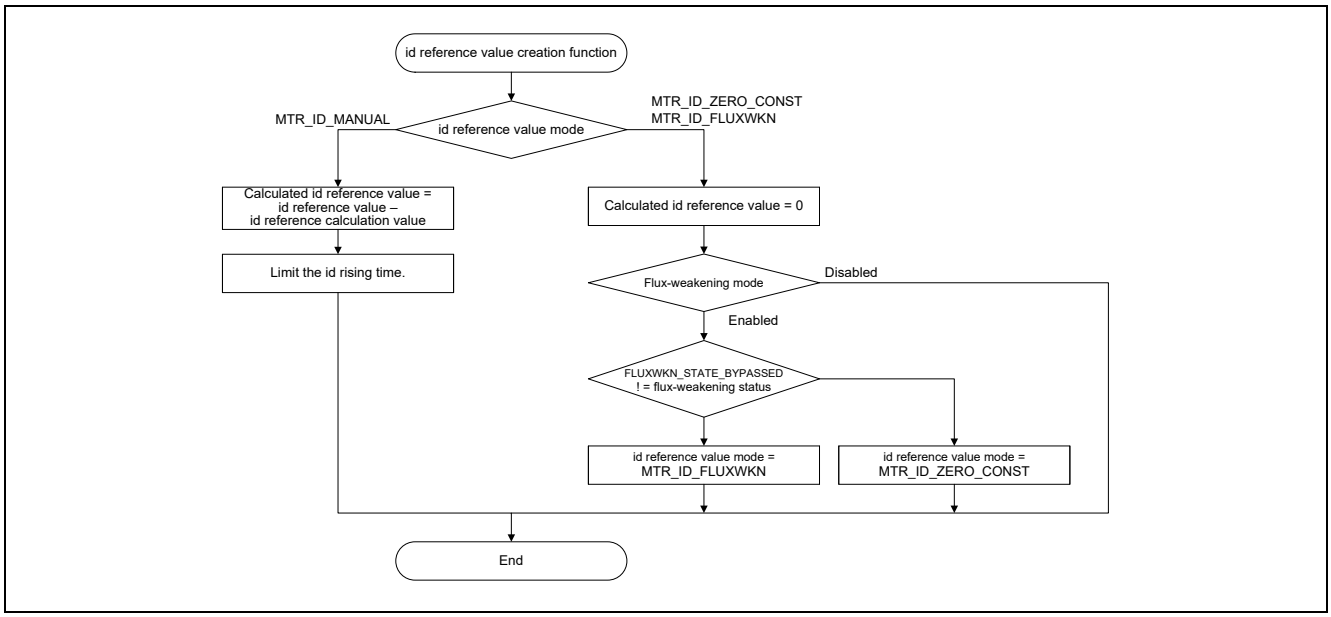
Position offset removal processing

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>float mtr_pos_offset_adjust (mtr_foc_control_t *st_foc); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | float | d-axis current reference value (A) | |
| Description | Removes the position offset. | | |
| Note | | | |

## mtr_set_id_ref

id reference value creation

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control.h"<br>float mtr_set_id_ref (mtr_foc_control_t *st_foc); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | float | d-axis current reference value (A) | |
| Description | Creates the d-axis current reference value. | | |
| Note | | | |



## mtr_set_iq_ref

iq reference value creation

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control .h"<br>float mtr_set_iq_ref ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | float | q-axis current reference value (A) | |
| Description | Creates the q-axis current reference value. | | |
| Note | | | |

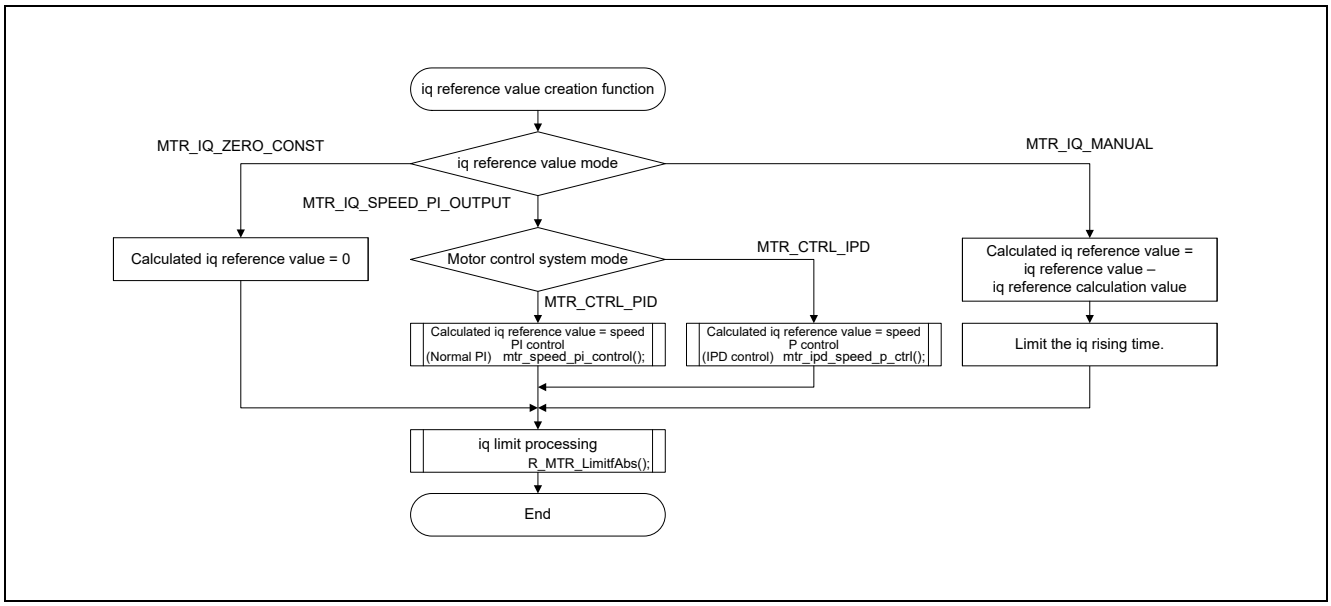


**Table 3.19 Speed Control Processing Functions**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_speed.c | mtr_speed_pi_control<br>Input: mtr_speed_control_t *st_sc:<br>Pointer to speed control structure<br>float f4_speed_rad: Speed<br>Output: float: q-axis current reference value (A) | Speed PI control |
| r_mtr_spm_rslv_foc_rx/src /functions/ r_mtr_ipd.obj | mtr_ipd_speed_p_ctrl<br>Input: mtr_ipd_ctrl_t *st_ipd:<br>Pointer to IPD control structure<br>float f4_ref_speed_rad:<br>Speed reference value<br>float f4_speed_rad: Speed<br>Output: float: q-axis current reference value (A) | Speed control block for IPD control |

## mtr_set_speed_ref

Speed reference value creation

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control .h"<br>float mtr_set_speed_ref ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | float | Speed reference value (rad/s) | |
| Description | Creates the peed reference value. | | |
| Note | | | |

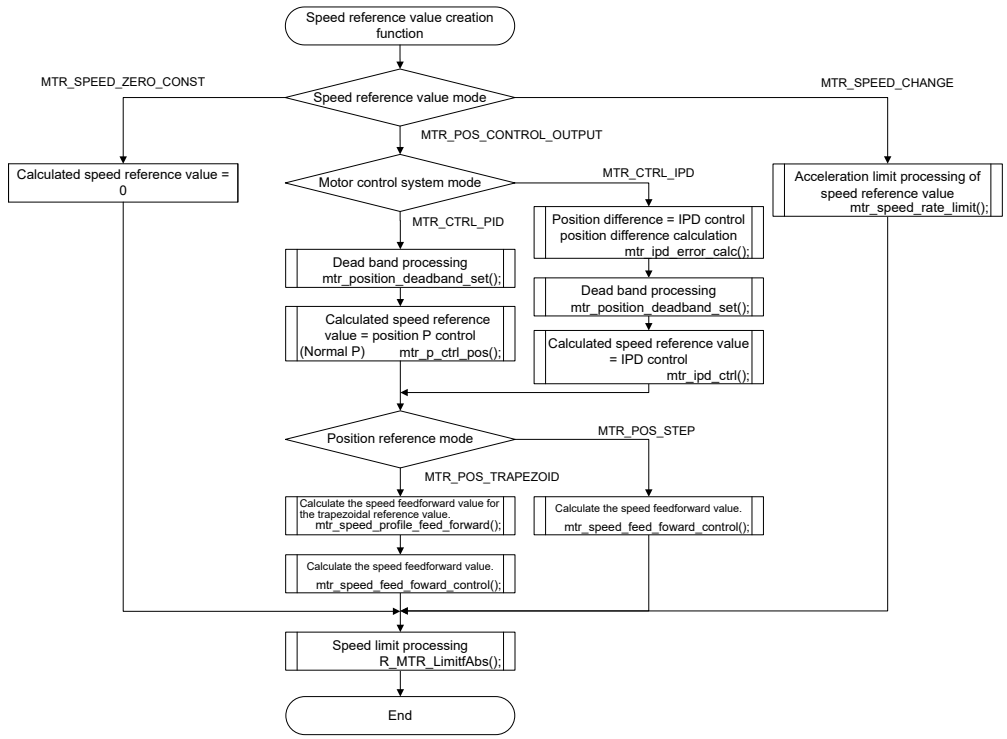


**Table 3.20 Speed Reference Value Creation Processing Functions**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_stm_rslv_control.c | mtr_position_deadband_set<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to vector control structure<br>float f4_pos_err_rad: Position difference<br>Output: float: Dead band processed difference (rad) | Returns the difference with a dead band. |
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_position.c | mtr_p_ctrl_pos<br>Input: mtr_position_control_t *st_pc:<br>Pointer to position control structure<br>mtr_parameter_t *st_motor:<br>Pointer to motor parameter structure<br>float f4_pos_err_rad: Position difference<br>Output: float: Speed reference value (rad/s) | Proportional position control |
| r_mtr_spm_rslv_foc_rx/src /functions/ r_mtr_ipd.obj | mtr_ipd_error_calc<br>Input: mtr_ipd_ctrl_t *st_ipd:<br>Pointer to IPD control structure<br>float f4_pos_rad: Position<br>float f4_ref_pos_rad_ctrl:<br>Position reference value<br>Output: float: Position difference (rad) | Calculates the IPD control position difference. |
| | mtr_ipd_ctrl<br>Input: mtr_ipd_ctrl_t *st_ipd:<br>Pointer to IPD control structure<br>mtr_parameter_t *st_motor:<br>Pointer to motor parameter structure<br>float f4_pos_err_rad: Position difference<br>float f4_max_speed_rad:<br>Maximum speed value<br>Output: float: Speed reference value (rad/s) | IPD control |
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_position_profiling.c | mtr_speed_profile_feed_forward<br>Input: mtr_position_profiling_t *st_pf:<br>Pointer to position reference creation structure<br>Output: float: Speed feedforward value in mechanical angle (rad/s]) | Calculates the speed feedforward value for trapezoidal reference. |
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_position.c | mtr_speed_feed_foward_control<br>Input: mtr_position_control_t *st_pc:<br>Pointer to position control structure<br>float f4_speed_ff_rad:<br>Speed feedforward value in mechanical angle<br>float f4_speed_ref_calc_rad:<br>Speed reference value<br>uint16_t u2_mtr_pp: Number of pole pairs<br>Output: float: Speed reference value (including feedforward value) (rad/s) | Adds the speed feedforward value to the speed reference. |
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_foc_speed.c | mtr_speed_rate_limit<br>Input: mtr_speed_control_t *st_sc:<br>Pointer to speed control structure<br>Output: float: Speed reference value (acceleration limit processing) (rad/s) | Acceleration limit processing |
| r_mtr_spm_rslv_foc_rx/src /functions/ r_mtr_filter.c | mtr_com_LimitifAbs<br>Input: float f4_value: Variable to be limited<br>float f4_limit_value: Limit value<br>Output: float: Limited variable value | Output limit processing |

## mtr_set_pos_ref

Position reference value creation

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control .h"<br>float mtr_set_pos_ref ( mtr_foc_control_t *st_foc ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| Return value | float | Position reference value (rad) | |
| Description | Creates a position reference value. | | |
| Note | | | |

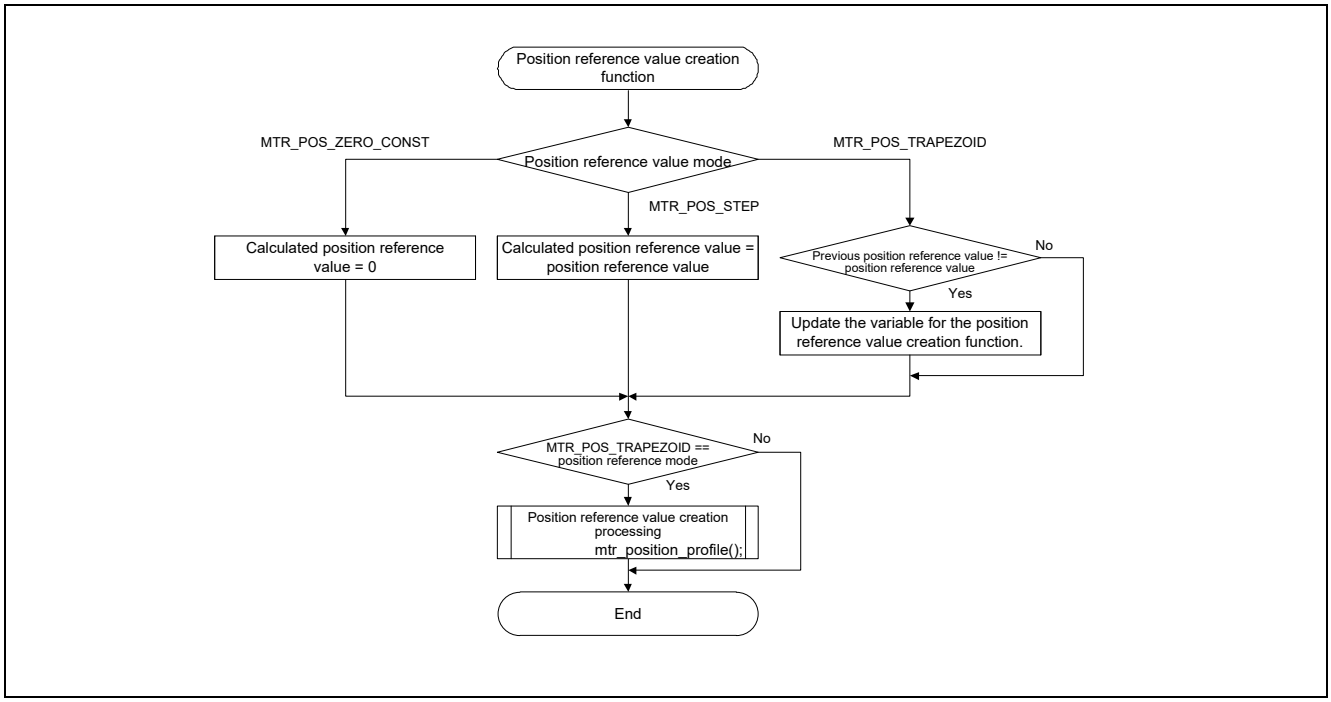



**Table 3.21 Position Reference Value Creation Processing Function**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src /controller/ r_mtr_position_profiling.c | mtr_position_profile<br>Input: mtr_position_profiling_t *st_pf:<br>Pointer to position reference creation structure<br>Output: float: Position reference value (rad) | Creates a trapezoidal reference value. |

## mtr_flux_weakng

Flux-weakening control execution

| | | | |
|---|---|---|---|
| Syntax | #include "r_mtr_foc_stm_rslv_control .h"<br>void mtr_flux_weakng ( mtr_foc_control_t *st_foc, float *f4_idq_ref ); | | |
| Argument | mtr_foc_control_t *st_foc | I/O | Pointer to the vector control structure |
| | float *f4_idq_ref | I/O | dq-axis current pointer |
| Return value | None | | |
| Description | Executes flux-weakening control for the dq-axis current reference value. | | |
| Note | | | |

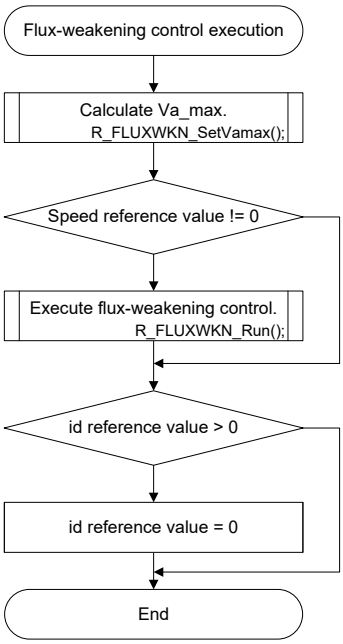


**Table 3.22 Flux-Weakening Control Processing Functions**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src /functions/ r_mtr_fluxwkn.obj | r_fluxwkn_SetVamax<br>Input: fluxwkn_t *p_fluxwkn:<br>Pointer to flux-weakening control structure<br>float f4_va_max: Maximum value of Va<br>Output: None | Calculates the maximum value of Va. |
| | mtr_fluxwkn_run<br>Input: fluxwkn_t *p_fluxwkn:<br>Pointer to flux-weakening control structure<br>float f4_speed_rad: Speed<br>float *p_f4_idq: dq-axis current pointer<br>float *p_f4_idq_ref:<br>Pointer to dq-axis current reference value<br>Output: uint16_t: Status of the module<br>FLUXWKN_STATE_BYPASSED<br>FLUXWKN_STATE_FLUXWKN<br>FLUXWKN_STATE_IDSAT<br>FLUXWKN_STATE_ERROR<br>FLUXWKN_STATE_INVALID_MOTOR<br>FLUXWKN_STATE_INVALID_IAMAX<br>FLUXWKN_STATE_INVALID_VAMAX<br>FLUXWKN_STATE_INVALID_VFWRATIO | Executes flux-weakening control. |

### 3.10.3 Acquiring Angle at the Time of Current Detection (mtu3_tciv4_interrupt())

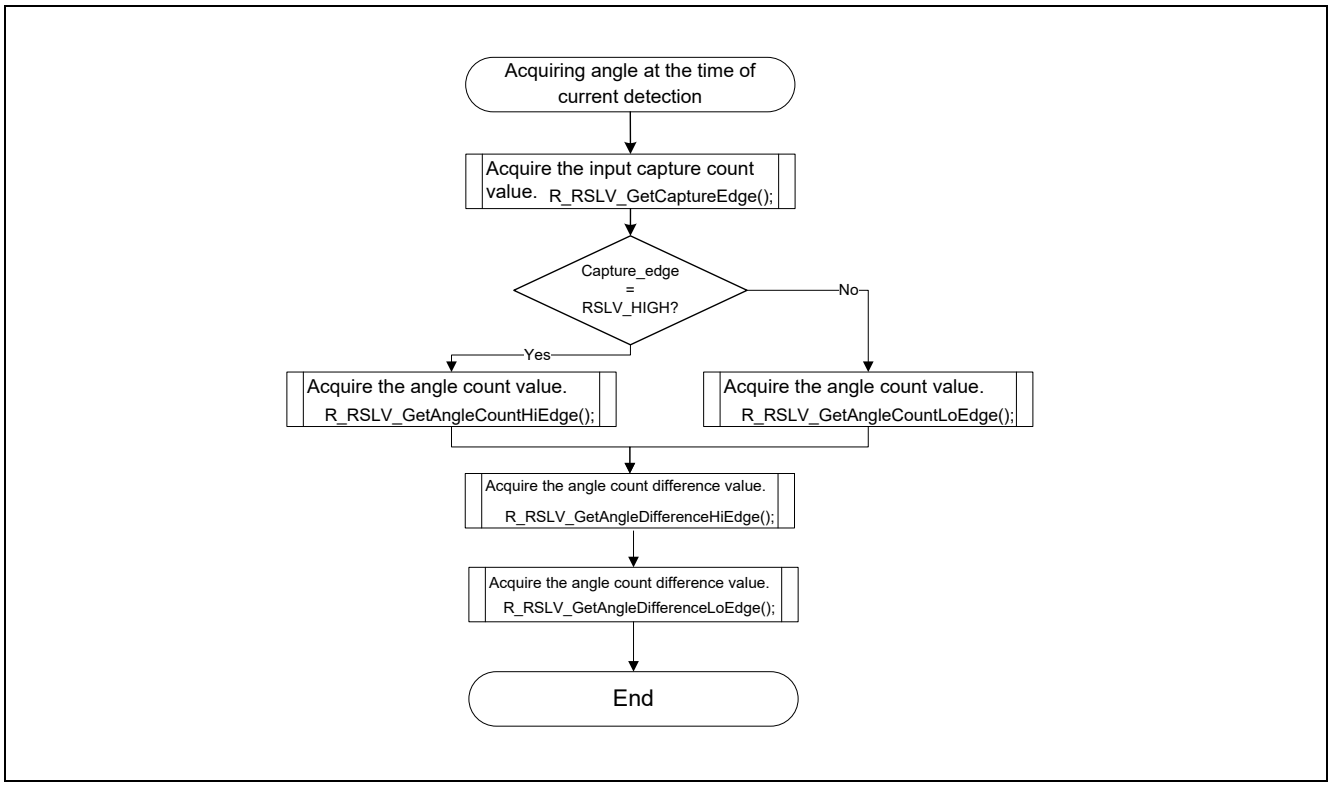



**Figure 3.26 Flow of Processing at the Time of Current Detection (in PWM Trough Interrupt)**

**Table 3.23 Functions for Acquiring Angle at the Time of Current Detection**

| File Name | Function | Processing |
|---|---|---|
| driver/sensor/rdc_driver_RX/api/r_rslv_driver_api.h | R_RSLV_GetCaptureEdge<br>Input: uint8_t *cap_edge<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Acquires the count value obtained by input capture.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_GetAngleCountHiEdge<br>Input: uint16_t *angle_cnt:<br>Pointer to variable for storing acquired count value<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Acquires the count value on the rising edge.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_GetAngleCountLoEdge<br>Input: uint16_t *angle_cnt:<br>Pointer to variable for storing acquired count value<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Acquires the count value on the falling edge.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_GetAngleDifferenceHiEdge<br>Input: uint16_t *angle_cnt:<br>Pointer to variable for storing acquired count value<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Acquires the count value difference between rising edges.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_GetAngleDifferenceLoEdge<br>Input: uint16_t *angle_cnt:<br>Pointer to variable for storing acquired count value<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Acquires the count value difference between falling edges.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |

### 3.10.4 Overcurrent Detection Interrupt Processing (R_MTR_SR_Foc_INT_OverCur())

An overcurrent detection interrupt is generated when a falling edge is detected on the POE0# pin as the high impedance control condition for the PWM output pin in the software covered by this document or when a short-circuit of an output is detected in output level comparison. For this reason, the PWM output pin is already in a high-impedance state at the beginning of this interrupt processing, and the output signal to the motor is stopped.

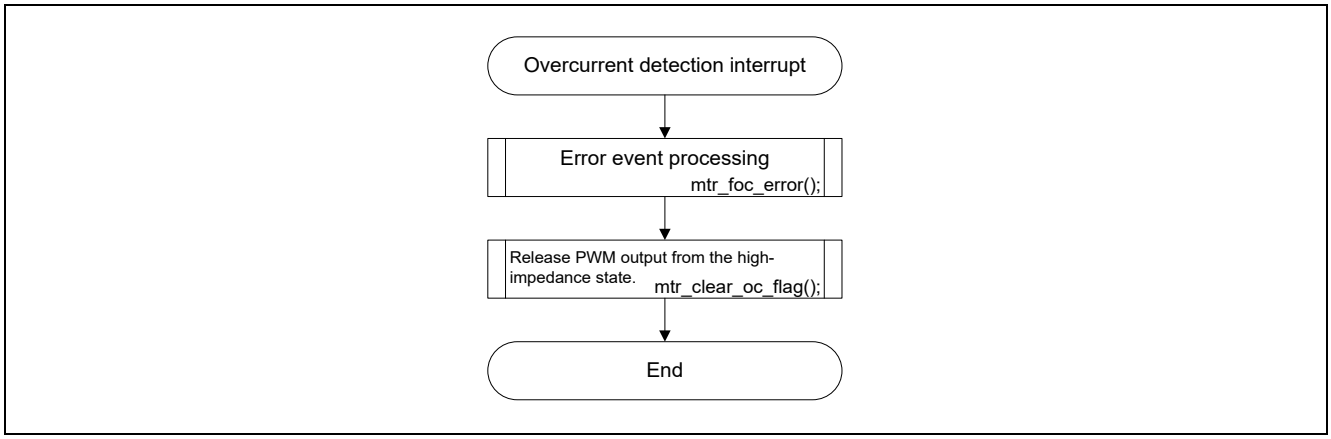



**Figure 3.27 Overcurrent Detection Interrupt Processing Flowchart**

**Table 3.24 Overcurrent Detection Interrupt Processing Functions**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src/controller/r_mtr_foc_stm_rslv_control.c | mtr_foc_error<br>Input: mtr_foc_control_t *st_foc:<br>Pointer to FOC structure<br>uint16_t u2_error_flag:<br>Error flag<br>Output: None | Generates an error event upon receiving the input error flag. |
| driver/mcu/r_mtr_ctrl_rx24t.c | mtr_clear_oc_flag<br>Input: None<br>Output: None | Clears the POE flag. |

### 3.10.5 Main Function Processing


**Figure 3.28 Main Processing Flowchart**

**Table 3.25 Main Processing Functions (1/2)**

| File Name | Function | Processing |
|---|---|---|
| driver/mcu/<br>r_mtr_ctrl_rx24t.c | R_MTR_InitHardware<br>Input: None<br>Output: None | Sets up initialization for peripheral functions of the MCU. |
| driver/sensor/<br>r_ctrl_rdc_driver_adapter.c | RESOLVER_peripheral_init<br>Input: None<br>Output: None | Initializes resolver-related variables. |
| | RDC_peripheral_init<br>Input: None<br>Output: None | Sets up initialization for RDC-related peripheral functions. |
| driver/sensor/rdc_driver_RX/<br>api/r_rslv_driver_api.h | R_RSLV_Rdc_VariableInit<br>Input: unsigned char *u1_init_dat:<br>RDC register initialization data<br>Output: uint8_t: Processing result<br>RSLV_RESULT_OK<br>RSLV_RESULT_NG | Initializes RDC-related variables.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| application/user_interface/ics/<br>ICS_RX24T.obj | ics2_init<br>Input: void * addr: Address pointer<br>uint8_t port:<br>Serial communications port<br>uint8_t level: Interrupt level<br>uint8_t brr:<br>Communications baud rate<br>uint8_t mode:<br>Communications mode<br>Output: None | Sets up initialization for RMW communications. |
| driver/mcu/<br>r_mtr_ctrl_rx24t.c | R_MTR_Start_s12ad<br>Input: None<br>Output: None | Starts A/D conversion. |
| | R_MTR_Start_cmt0<br>Input: None<br>Output: None | Starts the CMT0 timer. |
| | R_POE3_C8_Start<br>Input: None<br>Output: None | Starts POE3. |
| application/main/<br>main.c | variables_init<br>Input: None<br>Output: None | Initializes global variables. |
| r_mtr_spm_rslv_foc_rx/src/<br>r_mtr_stm_rslv_foc_rx.c | R_MTR_SR_Foc_Open<br>Input: uint8_t u1_id: Motor ID<br>Output: r_mtr_rslv_ret_t: Processing result<br>MTR_SUCCESS<br>MTR_ERR_BAD_ID | Initializes vector control. |
| | R_MTR_SR_Foc_InitVariables<br>Input: uint8_t u1_id: Motor ID<br>Output: r_mtr_rslv_ret_t: Processing result<br>MTR_SUCCESS<br>MTR_ERR_BAD_ID | Initializes variables for controlling the motor |
| application/user_interface/<br>flash/<br>r_flash_access.c | mtr_read_flashdt<br>Input: None<br>Output: None | Acquires data from the flash memory. |

**Table 3.25 Main Processing Functions (2/2)**

| File Name | Function | Processing |
|---|---|---|
| r_mtr_spm_rslv_foc_rx/src/<br>r_mtr_stm_rslv_foc_rx.c | R_MTR_SR_Foc_SetFlash2Ctrl<br>Input: uint8_t u1_id: Motor ID<br>mtr_flash_val *pst_val:<br>Pointer to structure for storing data to be reflected<br>Output: r_mtr_rslv_ret_t: Processing result<br>MTR_SUCCESS<br>MTR_ERR_BAD_ID | Reflects information read from the flash memory in the controller side. |
| | R_MTR_SR_Foc_ExecEvent<br>Input: uint8_t u1_id: Motor ID<br>uint8_t u1_event: Event name<br>Output: r_mtr_rslv_ret_t: Processing result<br>MTR_SUCCESS<br>MTR_ERR_BAD_ID | Generates an event. |
| driver/sensor/rdc_driver_RX/<br>api/r_rslv_driver_api.h | R_RSLV_ESig_Start<br>Input: None<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Starts outputting the excitation signal.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV _CSig_Start<br>Input: None<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Starts outputting the carrier correction signal.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV _CSig_Stop<br>Input: None<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Stops outputting the carrier correction signal.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| application/main/<br>main.c | ui_main<br>Input: None<br>Output: None | Handles user interface processing. |
| application/user_interface/<br>flash/<br>r_flash_access.c | memory_write<br>Input: None<br>Output: None | Writes control variable values to the flash memory. |
| driver/mcu/<br>r_mtr_ctrl_rx24t.c | R_ICU_Start_irq5<br>Input: None<br>Output: None | Starts external interrupts (RDC alarm signal). |
| driver/sensor/rdc_driver_RX<br>/api/<br>r_rslv_driver_api.c | R_RSLV_Rdc_Init_Sequence<br>Input: uint16_t *init_status:<br>Address of variable for acquiring the initialization result<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | RDC initialization processing<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_Rdc_Communication<br>Input: None<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | RDC communications processing<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| | R_RSLV_Phase_AdjUpdate<br>Input: None<br>Output: uint8_t: Processing result<br>(always RSLV_RESULT_OK) | Sets the duty cycles of phase adjustment PWM signals.<br>(See the application note *"Using the Driver for Resolver-to-Digital Converter Control"*.) |
| driver/mcu/<br>r_mtr_ctrl_rx24t.c | R_MTR_ClearWdt<br>Input: None<br>Output: None | Clears the watchdog timer. |

## 3.11 List of Structures

### 3.11.1 "r_mtr_spm_rslv_foc_rx" Folder

**Table 3.26 Structure Definitions in "r_mtr_stm_rslv_foc_rx_if.h" (1/2)**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_flash_val | float | f4_rslv_angle_offset_rad | |
| | float | f4_offset_ia | a-phase current offset value (A) |
| | float | f4_offset_ib | b-phase current offset value (A) |
| mtr_ctrl_input_t | uint8_t | u1_flag_angle_spl_comp_use | Flag indicating whether to enable the resolver position error compensation function |
| | uint8_t | u1_flag_cc_spl_comp_use | Flag indicating whether to enable the current control sampling delay compensation function |
| | uint8_t | u1_flag_bpf_delay_comp_use | Flag indicating whether to enable the BPF delay compensation function |
| | uint8_t | u1_flag_sob_use | Flag indicating whether to enable the speed observer |
| | uint8_t | u1_flag_volt_err_comp_use | Flag indicating whether to enable the voltage error compensation function |
| | uint8_t | u1_flag_flux_weakening_use | Flag indicating whether to enable the flux-weakening function |
| | uint8_t | u1_ctrl_loop_mode_buff | Control loop selection |
| | uint8_t | u1_ctrl_method_mode | Control method selection |
| | uint8_t | u1_state_pos_ref_buff | Position reference value management |
| | uint8_t | u1_offset_adjust_mode_buff | Offset adjustment mode |
| | uint8_t | u1_direction | Reference rotation direction |
| | int16_t | f4_ref_position_deg | Reference position (degree) |
| | int16_t | s2_ref_speed_rpm | Reference speed in mechanical angle (rpm) |
| | uint16_t | u2_min_speed_rpm | Minimum speed in mechanical angle (rpm) |
| | uint16_t | u2_max_speed_rpm | Maximum speed in mechanical angle (rpm) |
| | uint16_t | u2_speed_limit_rpm | Overspeed limit in mechanical angle (rpm) |
| | uint16_t | u2_pos_dead_band | Position dead band |
| | uint16_t | u2_pos_band_limit | Positioning band limit |
| | uint16_t | u2_encd_cpr_mech | Encoder count per revolution in mechanical angle |
| | uint16_t | u2_pos_interval_time | Interval time for reference position update |
| | uint16_t | u2_offset_calc_time | Calculation time for current offset / 100 (µs) |
| | uint16_t | u2_offset_calc_wait | Counter for current offset detection start timing |
| | uint16_t | u2_run_mode | Run mode |
| | float | f4_ipd_pos_kp_ratio | Proportional position control gain ratio for IPD control |
| | float | f4_ipd_speed_k_ratio | Speed gain ratio for IPD control |
| | float | f4_ipd_err_limit_1 | Position error limit 1 |
| | float | f4_ipd_err_limit_2 | Position error limit 2 |
| | float | f4_accel_time | Acceleration time |
| | float | f4_speed_rate_limit | Speed rate limit (rpm/ms) |
| | float | f4_ref_id | Motor d-axis current reference (A) |
| | float | f4_ref_iq | Motor q-axis current reference (A) |
| | float | f4_current_rate_limit | q-axis current rate limit (A/ms) |
| | float | f4_bpf_comp_gain_cw | Gain of BPF delay compensation for CW |
| | float | f4_bpf_comp_gain_ccw | Gain of BPF delay compensation for CCW |
| | float | f4_bpf_comp_base_cw | Base coefficient of BPF delay compensation for CW |
| | float | f4_bpf_comp_base_ccw | Base coefficient of BPF delay compensation for CCW |

**Table 3.26 Structure Definitions in “r_mtr_stm_rslv_foc_rx_if.h” (2/2)**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_ctrl_input_t | mtr_parameter_t | st_motor | Motor parameters |
| | mtr_design_parameter_t | st_design_params | Design parameters |
| | mtr_ctrl_gain_t | st_ctrl_gain | Control gain |
| r_stm_rslv_ctrl_option_t | uint8_t | u1_opt_rslv_spltm_comp | |
| | uint8_t | u1_opt_cc_spldly_comp | |
| | uint8_t | u1_opt_bpf_delay_comp | |
| | uint8_t | u1_opt_sob | |
| | uint8_t | u1_opt_volt_err_comp | |
| | uint8_t | u1_opt_flux_weakning | |

### 3.11.2 “r_mtr_spm_rslv_foc_rx/src/controller” Folder

**Table 3.27 Structure Definitions in “r_mtr_foc_stm_rslv_control.h” (1/3)**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_foc_control_t | uint8_t | u1_id | Motor ID |
| | uint8_t | u1_state_id_ref | d-axis current reference value state<br>0: Zero reference<br>1: Flux-weakening control<br>2: Manual reference setting |
| | uint8_t | u1_state_iq_ref | q-axis current reference value state<br>0: Zero reference<br>1: Speed PI output<br>2: Manual reference setting |
| | uint8_t | u1_state_speed_ref | Speed reference value state<br>0: Zero reference<br>1: Position control output<br>2: Manual reference setting |
| | uint8_t | u1_state_pos_ref | Position reference value state<br>0: Zero reference<br>1: Step reference<br>2: Trapezoidal reference |
| | uint8_t | u1_state_pos_ref_buff | Position state variable buffer |
| | uint8_t | u1_state_angle_adj | Resolver angle offset detection state |
| | uint8_t | u1_direction | Direction of rotation |
| | uint8_t | u1_ctrl_loop_mode | Control loop mode<br>0: d-axis current control<br>1: q-axis current control<br>2: Speed control<br>3: Position control |
| | uint8_t | u1_ctrl_method_mode | Control method mode<br>0: PID control (only proportion control is used in position control)<br>1: IPD control |
| | uint8_t | u1_offset_adjust_mode | Offset removal mode<br>0: Current offset<br>1: Position offset<br>2: Completed |
| | uint8_t | u1_offset_adjust_mode_buff | Offset removal mode buffer |
| | uint8_t | u1_flag_offset_calc | Offset calculation flag |
| | uint8_t | u1_flag_charge_cap | Power-supply voltage stable flag |
| | uint8_t | u1_flag_rdc_state_ready | RDC preparation completed flag |
| | uint8_t | u1_flag_angle_spl_comp_use | Magnetic pole position sampling delay compensation enable flag |
| | uint8_t | u1_flag_pos_delay_comp_use | Position sampling delay compensation enable flag |
| | uint8_t | u1_flag_cc_spl_comp_use | Voltage reflection timing angle correction enable flag |
| | uint8_t | u1_flag_bpf_delay_comp_use | RDC bandpass filter delay compensation enable flag |
| | uint8_t | u1_flag_sob_use | Speed observer enable flag |
| | uint8_t | u1_flag_volt_err_comp_use | Voltage error compensation enable flag |
| | uint8_t | u1_flag_flux_weakening_use | Flux-weakening control enable flag |
| | uint8_t | u1_flag_mode_inactive | Drive mode inactive flag |

**Table 3.27 Structure Definitions in "r_mtr_foc_stm_rslv_control.h" (2/3)**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_foc_control_t | uint16_t | u2_error_status | Error state (descriptions of individual values are omitted) |
| | uint16_t | u2_run_mode | Motor drive mode<br>0: Offset removal<br>1: Motor drive |
| | uint16_t | u2_offset_calc_time | Offset calculation time |
| | uint16_t | u2_offset_calc_wait | Offset calculation wait time |
| | uint16_t | u2_cnt_adjust | Offset calculation time counter |
| | uint16_t | u2_angle_adj_time | Position offset calculation time |
| | uint16_t | u2_angle_adj_cnt | Position offset calculation time counter |
| | uint16_t | u2_rslv_angle_cnt | Resolver electrical angle counter |
| | uint16_t | u2_rslv_pre_angle_cnt | Resolver electrical angle counter |
| | uint16_t | u2_rslv_timer_cnt | Resolver excitation timer count value |
| | uint16_t | u2_rslv_angle_offset_cnt | Resolver offset count value |
| | int32_t | s4_rslv_cycle_cnt | Resolver one-cycle counter |
| | int16_t | s2_angle_err_cnt | Resolver cycle lead angle count value |
| | float | f4_vdc_ad | Power-supply voltage |
| | float | f4_offset_ia | α-axis current offset (A) |
| | float | f4_offset_ib | β-axis current offset (A) |
| | float | f4_sum_ia_ad | Value to be added for calculating α-axis current offset |
| | float | f4_sum_ib_ad | Value to be added for calculating β-axis current offset |
| | float | f4_ia_ad | α-axis current (A) |
| | float | f4_ib_ad | β-axis current (A) |
| | float | f4_ref_va | α-axis reference voltage (V) |
| | float | f4_ref_vb | β-axis reference voltage (V) |
| | float | f4_moda | α-axis modulation rate |
| | float | f4_modb | β-axis modulation rate |
| | float | f4_overcurrent_limit | Overcurrent limit value (A) |
| | float | f4_overvoltage_limit | Overvoltage limit value (V) |
| | float | f4_undervoltage_limit | Low-voltage limit value (V) |
| | float | f4_overspeed_limit_rad | Overspeed limit value (rad) |
| | float | f4_spl_comp_angle_rad | Sampling compensation angle (rad) |
| | float | f4_spl_delay_time | Position sampling delay time |
| | float | f4_spl_delay_comp_pos_rad | Amount of advanced position due to position sampling delay (rad) |
| | float | f4_rslv_pos_rad | Resolver sensor position (rad) in mechanical angle |
| | float | f4_rslv_angle_rad | Resolver sensor angle (rad) in electrical angle |
| | float | f4_rslv_angle_offset_rad | Offset from the magnetic pole position of the resolver sensor (rad) |
| | float | f4_rslv_angle_offset_sum_rad | Total offset value |
| | float | f4_rslv_speed_rad | Resolver sensor speed (rad/s) in electrical angle |
| | float | f4_rslv_speed_rad_lpf | Resolver sensor speed after passing through the low-pass filter (rad/s) in electrical angle |
| | float | f4_open_loop_pos_rad | Open-loop angle (rad) |

**Table 3.27 Structure Definitions in "r_mtr_foc_stm_rslv_control.h" (3/3)**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_foc_control_t | mod_t | st_mod | Definition of the modulation module structure |
| | mtr_rotor_angle_t | st_rotor_angle | Definition of the motor angle management module structure |
| | mtr_statemachine_t | st_stm | Definition of the state machine module structure |
| | mtr_parameter_t | st_motor | Definition of the motor parameter module structure |
| | mtr_current_control_t | st_cc | Definition of the current control module structure |
| | mtr_speed_control_t | st_sc | Definition of the speed control module structure |
| | mtr_position_control_t | st_pc | Definition of the position control module structure |
| | mtr_encoder_t | st_ec | Definition of the encoder module structure |
| | mtr_position_profiling_t | st_ppf | Definition of the position profiling module structure |
| | mtr_ipd_ctrl_t | st_ipd | Definition of the IPD controller module structure |
| | mtr_speed_observer_t | st_sob | Definition of the speed observer module structure |
| | mtr_volt_comp_t | st_volt_comp_p | Definition of the voltage error compensation module structure |
| | fluxwkn_t | st_fluxwkn | Definition of the flux-weakening module structure |
| | mtr_1st_order_lpf_t | st_slpf | Definition of the first-order low-pass filter module structure |
| | rslv_compensation_t | st_rcomp | Definition of the resolver angle correction structure |

**Table 3.28 Structure Definitions in "r_mtr_foc_current.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_current_control_t | float | f4_ctrl_period | Control cycle (s) |
| | float | f4_ref_vd | d-axis reference voltage |
| | float | f4_ref_vq | q-axis reference voltage |
| | float | f4_ref_id | d-axis reference current (reference setting from the application layer) |
| | float | f4_ref_iq | q-axis reference current (reference setting from the application layer) |
| | float | f4_ref_id_ctrl | d-axis reference current (input to the control loop) |
| | float | f4_ref_iq_ctrl | q-axis reference current (input to the control loop) |
| | float | f4_current_rate_limit | Current increasing speed limit value |
| | float | f4_id_ad | Detected d-axis current value |
| | float | f4_iq_ad | Detected q-axis current value |
| | float | f4_pre_id_ad | Previous value of d-axis current (A) |
| | float | f4_pre_iq_ad | Previous value of q-axis current (A) |
| | float | f4_lim_iq | q-axis current limit |
| | mtr_pi_ctrl_t | st_pi_id | Structure of PI control variables of the d-axis current control loop |
| | mtr_pi_ctrl_t | st_pi_iq | Structure of PI control variables of the q-axis current control loop |

**Table 3.29 Structure Definitions in "r_mtr_foc_speed.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_speed_control_t | uint8_t | u1_ref_dir | Direction of rotation |
| | float | f4_rpm_rad | Coefficient of conversion from rpm to electrical angle (rad) |
| | float | f4_speed_ctrl_period | Speed control cycle |
| | float | f4_ref_speed_rad_ctrl | Speed reference value input to the control loop (rad) |
| | float | f4_ref_speed_rad | Speed reference value from the application layer (rad) |
| | float | f4_speed_rad | Detected speed received from the sensor (rad) |
| | float | f4_speed_rad_ctrl | Detected speed after filtering (rad) |
| | float | f4_max_speed_rad | Maximum speed |
| | float | f4_speed_rate_limit | Speed limit value |
| | mtr_pi_ctrl_t | st_pi_speed | Structure of PI control variables of the speed control loop |

**Table 3.30 Structure Definitions in "r_mtr_foc_position.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_position_control_t | uint16_t | u2_pos_dead_band | Dead band of position (± count) |
| | uint16_t | u2_pos_band_limit | Positioning completion range (± count) |
| | float | f4_pos_kp | Proportional gain of position |
| | float | f4_pos_err_rad | Position difference |
| | float | f4_pos_rad | Detection position |
| | float | f4_ref_pos_rad | Position reference value |
| | float | f4_ref_pos_pre_rad | Previous value of position reference |
| | float | f4_ref_pos_rad_ctrl | Position reference value input to the control loop |
| | float | f4_speed_ff_rad | Speed feedforward value (rad/s) |
| | float | f4_speed_ff_ratio | Coefficient of speed feedforward value |

**Table 3.31 Structure Definitions in "r_mtr_position_profiling.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_position_profiling_t | uint8_t | u1_state_pos_pf | Position profiling function status |
| | uint8_t | u1_pos_ref_mode | Position profiling mode |
| | uint16_t | u2_interval_time | Position reference update interval time |
| | uint16_t | u2_interval_time_buff | Buffer for position reference update interval time |
| | uint16_t | u2_interval_time_cnt | Counter of position reference update interval time |
| | float | f4_accel_time | Acceleration time |
| | float | f4_accel_time_buff | Acceleration time buffer |
| | float | f4_accel_time_inv | Reciprocal of acceleration time |
| | float | f4_max_accel_time | Maximum acceleration time |
| | float | f4_accel_max_speed | Maximum speed |
| | float | f4_accel_max_speed_buff | Maximum speed buffer |
| | float | f4_time_sec | Time management variable (s) |
| | float | f4_pos_st_rad | Start position of position profiling(rad) |
| | float | f4_pos_ed_rad | Stop position of position profiling (rad) |
| | float | f4_pos_dt_rad | Position difference (rad) |
| | float | f4_pos_dt_time_sec | Position arrival time (s) |

**Table 3.32 Structure Definitions in "r_mtr_pi_control.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_pi_ctrl_t | float | f4_err | Difference |
| | float | f4_kp | Proportional gain |
| | float | f4_ki | Integral gain |
| | float | f4_refi | Calculated integral value |
| | float | f4_ilimit | PI output limit value |

**Table 3.33 Structure Definitions in "r_mtr_transform.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_rotor_angle_t | float | f4_rotor_angle_rad | Rotor angle (rad) (electrical angle) |
| | float | f4_sin | Calculated sine value according to rotor angle |
| | float | f4_cos | Calculated cosine value according to rotor angle |

**Table 3.34 Structure Definitions in "r_mtr_parameter.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_parameter_t | uint16_t | u2_mtr_pp | Pole pairs |
| | float | f4_mtr_r | Resistance (ohm) |
| | float | f4_mtr_ld | Inductance of d-axis (H) |
| | float | f4_mtr_lq | Inductance of q-axis (H) |
| | float | f4_mtr_m | Magnetic flux (Wb) |
| | float | f4_mtr_j | Rotor inertia (kgm^2) |
| | float | f4_nominal_current_rms | Rated torque (Nm) |
| mtr_design_parameter_t | float | f4_current_omega | Natural frequency (Hz) for current loop gain design |
| | float | f4_current_zeta | Damping ratio for current loop gain design |
| | float | f4_speed_omega | Natural frequency (Hz) for speed loop gain design |
| | float | f4_speed_zeta | Damping ratio for speed loop gain design |
| | float | f4_speed_lpf_omega | Natural frequency (Hz) for speed lpf gain design |
| | float | f4_pos_omega | Natural frequency (Hz) for position loop gain design |
| | float | f4_sob_omega | Natural frequency (Hz) for speed observer gain design |
| | float | f4_sob_zeta | Damping ratio for speed observer gain design |
| mtr_ctrl_gain_t | float | f4_id_kp | P gain of d-axis current PI for manual gain setting |
| | float | f4_id_ki | I gain of d-axis current PI for manual gain setting |
| | float | f4_iq_kp | P gain of q-axis current PI for manual gain setting |
| | float | f4_iq_ki | I gain of q-axis current PI for manual gain setting |
| | float | f4_speed_kp | P gain of speed PI for manual gain setting |
| | float | f4_speed_ki | I gain of speed PI for manual gain setting |
| | float | f4_pos_kp | P gain of position P for manual gain setting |

### 3.11.3 “r_mtr_spm_rslv_foc_rx/src/functions” Folder

**Table 3.35 Structure Definitions in “r_mtr_filter.h”**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_2nd_order_lpf_t | float | f4_pre_output | Previous output value |
| | float | f4_pre2_output | Previous output value but one |
| | float | f4_pre_input | Previous input value |
| | float | f4_pre2_input | Previous input value but one |
| | float | f4_omega_t | Coefficient ωt |
| | float | f4_omega2_t | Coefficient ω^2t |
| | float | f4_omega2_t2 | Coefficient ω^2t^2 |
| | float | f4_gain_ka | Gain Ka |
| | float | f4_gain_kb | Gain Kb |
| | float | f4_gain_kc | Gain Kc |
| mtr_1st_order_lpf_t | float | f4_pre_output | Previous output value |
| | float | f4_pre_input | Previous input value |
| | float | f4_omega_t | Coefficient ωt |
| | float | f4_gain_ka | Gain Ka |
| | float | f4_gain_kb | Gain Kb |

**Table 3.36 Structure Definitions in “r_mtr_mod.h”**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mod_t | float | f4_vdc | Power-supply voltage (V) |
| | float | f4_1_div_vdc | Reciprocal of power-supply voltage |
| | float | f4_voltage_error_ratio | Ratio of dead time to PWM cycle |
| | float | f4_max_duty | Maximum duty value |
| | float | f4_min_duty | Minimum duty value |
| | float | f4_neutral_duty | 0-V duty |
| | uint8_t | u1_sat_flag | Voltage saturation flag |

**Table 3.37 Structure Definitions in “r_mtr_volt_err_comp.h”**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_volt_comp_t | float | f4_comp_v [ARY_SIZE] | Vertical-axis voltage compensation value in the voltage error compensation table (V) |
| | float | f4_comp_i [ARY_SIZE] | Horizontal-axis current in the voltage error compensation table (A) |
| | float | f4_slope [ARY_SIZE +1] | Previous value of calculated torque (N▸m) |
| | float | f4_intcept [ARY_SIZE +1] | Linear approximation intercept of voltage error compensation |
| | float | f4_volt_comp_array [3] | Voltage compensation value |
| | float | f4_vdc | DC bus voltage value (V) |
| | float | f4_volt_comp_limit | Limit of voltage compensation value (V) |
| | uint8_t | u1_volt_err_comp_enable | Voltage error compensation enable flag |

**Table 3.38 Structure Definitions in "r_mtr_fluxwkn.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| fluxwkn_t | const mtr_parameter_t | *pmotor | Pointer to the motor parameter structure |
| | float | f4_ia_max | Absolute value of maximum current vector |
| | float | f4_va_max | Absolute value of maximum voltage vector |
| | float | f4_vfw_ratio | Use rate of flux-weakening control voltage |
| | float | f4_id_demag | Magnetic flux divided by d-axis inductance |
| | float | f4_id_min | Minimum d-axis current limit value |
| | float | f4_v_fw | Voltage used for flux-weakening control |
| | uint16_t | u2_fw_status | Flux-weakening control status |

**Table 3.39 Structure Definitions in "r_rslv_compensation.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| rslv_compensation_t | uint16_t | u2_rslv_timer_cnt_capture | Resolver input capture count value |
| | uint16_t | u2_rslv_pre_timer_cnt_capture | Previous value of resolver input capture count |
| | uint16_t | u2_rslv_timer_cnt_capture_offset | Offset value for the resolver input capture counter; that is, the counted value indicating the difference between the position detected by the resolver and the phase of the magnetic pole of the motor when the resolver is connected |
| | int16_t | s2_timer_cnt_err | Resolver delay angle count value |
| | float | f4_time | Resolver correction time |
| | float | f4_bpf_comp_rad | Bandpass filter delay correction angle (rad) in electrical angle |
| | float | f4_angle_comp_rad | Resolver correction angle (rad) in electrical angle |
| | float | f4_pre_angle_rad | Previous angle value (rad) in electrical angle |
| | float | f4_over_corr_comp_angle_rad | Overcompensated correction angle (rad) |
| | float | f4_bpf_comp_gain_cw | Compensation adjustment coefficient (CW and CCW) |
| | float | f4_bpf_comp_gain_ccw | Compensation amount = base compensation coefficient × compensation adjustment coefficient |
| | float | f4_bpf_comp_base_cw | Base compensation coefficient (CW) |
| | float | f4_bpf_comp_base_ccw | Base compensation coefficient (CCW) |

**Table 3.40 Structure Definitions in "r_mtr_ipd.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_ipd_ctrl_t | uint8_t | u1_ipd_lpf_flag | 0: LPF disabled (default) 1: LPF enabled |
| | float | f4_ref_pos_pre_rad_ctrl | Previous position value for control in mechanical angle (rad) |
| | float | f4_ipd_pos_k | Position control gain for IPD |
| | float | f4_ipd_pos_1st_fb_rad | Feedback position in mechanical angle (rad) |
| | float | f4_ipd_pos_1st_fb_pre_rad | Previous value of feedback position in mechanical angle (rad) |
| | float | f4_ipd_pos_2nd_fb_rad | Feedback position in mechanical angle (rad) |
| | float | f4_ipd_ref_pos_rad | IPD control reference in mechanical angle (rad) |
| | float | f4_ipd_err_rad | IPD control error in mechanical angle (rad) |
| | float | f4_ipd_pos_fb_k | Feedback gain for IPD |
| | float | f4_ipd_pos_ff_rad | Feedforward position in mechanical angle (rad) |
| | float | f4_ipd_pos_ff_k | Feedforward position control gain |
| | float | f4_ipd_pos_p_rad | Proportional control in mechanical angle (rad) |
| | float | f4_ipd_pos_kp | Position control gain kp |
| | float | f4_ipd_pos_kp_ratio | Position control gain kp ratio |
| | float | f4_ipd_pos_ff_ratio | Position feedforward gain ratio |
| | float | f4_ipd_speed_k | Speed gain for IPD |
| | float | f4_ipd_speed_k_ratio | Speed gain for IPD gain ratio |
| | float | f4_ipd_ref_speed_rad | Reference speed in electrical angle (rad/s) |
| | float | f4_ipd_err_limit_1 | Position error limit 1 |
| | float | f4_ipd_err_limit_2 | Position error limit 2 |
| | float | f4_ipd_lpf_omega | Natural frequency of position LPF |
| | float | f4_ipd_lpf_zeta | Damping ratio for position LPF |
| | mtr_2nd_order_lpf_t | st_pos_lpf | Second-order LPF structure |

**Table 3.41 Structure Definitions in "r_mtr_speed_observer.h"**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_speed_observer_t | float | f4_speed_rad | Speed observer output speed in electrical angle (rad/s) |
| | float | f4_ref_torque | Reference torque |
| | float | f4_ref_pre_torque | Previous value of reference torque |
| | float | f4_ref_speed_rad | Reference speed for the speed observer (SOB) in electrical angle (rad/s) |
| | float | f4_ref_pre_speed_rad | Previous value of reference speed for SOB in electrical angle (rad/s) |
| | float | f4_hpf_k1 | HPF gain 1 for SOB |
| | float | f4_hpf_k2 | HPF gain 2 for SOB |
| | float | f4_hpf_k3 | HPF gain 3 for SOB |
| | float | f4_k1 | K1 gain for SOB |
| | float | f4_k2 | K2 gain for SOB |
| | float | f4_hpf_ref_speed_rad | HPF output reference speed (rad/s) |
| | float | f4_hpf_ref_pre_speed_rad | Previous value of HPF output reference speed (rad/s) |
| | float | f4_hpf_omega | Natural frequency of speed observer HPF |
| | mtr_2nd_order_lpf_t | st_lpf | Second-order LPF structure |

### 3.11.4 “r_mtr_spm_rslv_foc_rx/src/mode” Folder

**Table 3.42 Structure Definitions in “r_mtr_statemachine.h”**

| Structure Name | Type | Variable Name | Remark |
|---|---|---|---|
| mtr_statemachine_t | uint8_t | u1_status | Current system status |
| | uint8_t | u1_status_next | Next system status to be entered |
| | uint16_t | u2_error_status | State transition error management |
| | uint8_t | u1_current_event | Current event |

## 3.12 Software Macro Definitions

### 3.12.1 “r_mtr_spm_rslv_foc_rx” Folder

**Table 3.43 Macro Definitions in “r_mtr_stm_rslv_foc_rx_if.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_motor_parameter.h | MTR_SPM_RSLV_FOC_VERSION_MAJOR | 1 | |
| | MTR_SPM_RSLV_FOC_VERSION_MINOR | 00 | |
| | MTR_GAIN_DESIGN_MODE | 0 | Using values in design_parameters to calculate gains in PI controllers |
| | MTR_GAIN_DIRECT_MODE | 1 | Accessing PI controllers directly |

### 3.12.2 “r_mtr_spm_rslv_foc_rx/ref” Folder

**Table 3.44 Macro Definitions in “r_mtr_motor_parameter.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_motor_parameter.h | MP_POLE_PAIRS | 50 | Number of pole pairs |
| | MP_MAGNETIC_FLUX | 0.0043f | Magnetic flux (Wb) |
| | MP_RESISTANCE | 1.2f | Resistance (Ω) |
| | MP_D_INDUCTANCE | 0.0027f | d-axis inductance (H) |
| | MP_Q_INDUCTANCE | 0.0027f | q-axis inductance (H) |
| | MP_ROTOR_INERTIA | 0.0000075f | Inertia (kgm^2) |
| | MP_NOMINAL_CURRENT_RMS | 1.414f | Rated current (Arms) |

**Table 3.45 Macro Definitions in “r_mtr_control_parameter.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_control_parameter.h | CP_POS_OMEGA | 10.0f | Natural frequency of position control loop (Hz) |
| | CP_SPEED_OMEGA | 40.0f | Natural frequency of speed control loop (Hz) |
| | CP_SPEED_ZETA | 1.0f | Attenuation coefficient for speed control loop |
| | CP_SPEED_LPF_OMEGA | 250.0f | Natural frequency of speed LPF |
| | CP_CURRENT_OMEGA | 400.0f | Natural frequency of current control loop (Hz) |
| | CP_CURRENT_ZETA | 1.0f | Attenuation coefficient for current control loop |
| | CP_SOB_OMEGA | 200.0f | Natural frequency of speed observer (Hz) |
| | CP_SOB_ZETA | 1.0f | Attenuation coefficient for speed observer |
| | CP_MIN_SPEED_RPM | 0 | Minimum speed in mechanical angle (rpm) |
| | CP_MAX_SPEED_RPM | 3000 | Maximum speed in mechanical angle (rpm) |
| | CP_SPEED_LIMIT_RPM | 4000 | Speed limit value in mechanical angle (rpm) |
| | CP_OL_ID_REF | 1.8f | d-axis current reference value (A) |

**Table 3.46 Macro Definitions in "r_mtr_inverter_parameter.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| inverter_parameter.h | IP_DEADTIME | 0.5f | Dead time (μs) |
| | IP_CURRENT_RANGE | 10.0f | A/D conversion range for current(A) (p-p value) |
| | IP_VDC_RANGE | 111.383f | A/D conversion range for inverter bus voltage (V) |
| | IP_INPUT_V | 24.0f | Inverter input voltage (V) |
| | IP_CURRENT_LIMIT | 5.0f | Overcurrent limit value (A) * |
| | IP_OVERVOLTAGE_LIMIT | 40.0f | Overvoltage limit value (V) |
| | IP_UNDERVOLTAGE_LIMIT | 18.0f | Low-voltage limit value (V) |

Note: This value is calculated from the rated power of the shunt resistor.

**Table 3.47 Macro Definitions in "r_resolver_rdc_command.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_resolver_rdc_command.h | RDC_CMD_INITIAL_VALUE_02h | 0x01 | Initial value of the RDC register at the specified address |
| | RDC_CMD_INITIAL_VALUE_04h | 0xDD | Initial value of the RDC register at the specified address |
| | RDC_CMD_INITIAL_VALUE_0Ah | 0xFF | Initial value of the RDC register at the specified address |
| | RDC_CMD_INITIAL_VALUE_16h | 0x11 | Initial value of the RDC register at the specified address |
| | RDC_CMD_INITIAL_VALUE_2Eh | 0x01 | Initial value of the RDC register at the specified address |
| | RDC_CMD_INITIAL_VALUE_42h | 0x80 | Initial value of the RDC register at the specified address |

**Table 3.48 Macro Definitions in "r_mtr_config.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_config.h | MP_MINEBEA_R17PMK440CNVA4438 | — | Motor selection |
| | CP_MINEBEA_R17PMK440CNVA4438 | — | |
| | RISE_EDGE_ONLY | — | Using only rising edges to read input capture value from RDC |
| | USE_FLASH | — | Enabling read and write access to flash memory |
| | USE_VOLT_ERR_COMP | 1 | Voltage error compensation<br>0: Disable 1: Enable |
| | USE_FLUX_WEAKENING | 1 | Flux-weakening control<br>0: Disable 1: Enable |
| | USE_RSLV_SPLTM_COMP | 1 | Sampling delay compensation of resolver position information<br>0: Disable 1: Enable |
| | USE_CC_SPLDLY_COMP | 1 | Correction of angle delay until reflection of voltage of the current control loop<br>0: Disable 1: Enable |
| | USE_BPF_DELAY_COMP | 1 | Delay correction by the bandpass filter of RDC<br>0: Disable 1: Enable |
| | USE_SOB | 1 | Speed observer<br>0: Disable 1: Enable |
| | POS_CTRL_MODE | MTR_CTRL_PID | Position control mode selection<br>MTR_CTRL_PID: PID controller<br>MTR_CTRL_IPD: IPD controller |
| | LOOP_MODE | MTR_LOOP_SPEED | Control loop mode selection<br>MTR_LOOP_CURRENT: Current loop mode<br>MTR_LOOP_SPEED: Speed loop mode<br>MTR_LOOP_POSITION: Position loop mode |
| | GAIN_MODE | MTR_GAIN_DESIGN_MODE | Gain mode<br>MTR_GAIN_DESIGN_MODE:<br>PI gain design mode<br>MTR_GAIN_DIRECT_MODE:<br>Direct PI gain input mode |
| | MOD_METHOD | MOD_METHOD_SPWM | Modulation method<br>MOD_METHOD_SPWM:<br>Sine wave modulation<br>MOD_METHOD_SVPWM:<br>Space vector modulation |
| | OFFSET_ADJUST_MODE | MTR_OFAJ_CURRENT_OFFSET | Offset detection mode<br>MTR_OFAJ_CURRENT_OFFSET: Current<br>MTR_OFAJ_POS_OFFSET: Position<br>MTR_OFAJ_FIN: Offset removal completed |
| | BPF_COMP_GAIN_BASE_CW | 4.75E-5f | Base gain for BPF compensation in clockwise rotation |
| | BPF_COMP_GAIN_BASE_CCW | 3.44E-05 | Base gain for BPF compensation in counterclockwise rotation |

### 3.12.3 “r_mtr_spm_rslv_foc_rx/src/controller” Folder

**Table 3.49 Macro Definitions in “r_mtr_foc_stm_rslv_control.h” (1/2)**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_foc_stm_rslv_control.h | MTR_ID_ZERO_CONST | 0 | d-axis current reference value set to zero |
| | MTR_ID_FLUXWKN | 1 | Flux-weakening control |
| | MTR_ID_MANUAL | 2 | Manual reference setting |
| | MTR_IQ_ZERO_CONST | 0 | q-axis current reference value set to zero |
| | MTR_IQ_SPEED_PI_OUTPUT | 1 | Using the speed PI output for reference |
| | MTR_IQ_MANUAL | 2 | Manual reference setting |
| | MTR_SPEED_ZERO_CONST | 0 | Speed reference value set to zero |
| | MTR_POS_CONTROL_OUTPUT | 1 | Using the position control output for reference |
| | MTR_SPEED_MANUAL | 2 | Manual reference setting |
| | MTR_POS_ZERO_CONST | 0 | Position reference value set to zero |
| | MTR_POS_STEP | 1 | Position step reference |
| | MTR_POS_TRAPEZOID | 2 | Position profiling reference |
| | MTR_LOOP_CURRENT_ID | 0 | d-axis current control mode |
| | MTR_LOOP_CURRENT_IQ | 1 | q-axis current control mode |
| | MTR_LOOP_SPEED | 2 | Speed control mode |
| | MTR_LOOP_POSITION | 3 | Position control mode |
| | MTR_CTRL_PID | 0 | Proportional position control |
| | MTR_CTRL_IPD | 1 | IPD control |
| | MTR_OFAJ_CURRENT_OFFSET | 0 | Current offset removal mode |
| | MTR_OFAJ_POS_OFFSET | 1 | Position offset removal mode |
| | MTR_OFAJ_FIN | 2 | Offset removal completed |
| | MTR_ANGLE_ADJ_90DEG | 0 | Position offset removal (alignment at 90 degrees) |
| | MTR_ANGLE_ADJ_0DEG | 1 | Position offset removal (alignment at 0 degrees) |
| | MTR_ANGLE_ADJ_FIN | 2 | Alignment completed |
| | MTR_MODE_OFFSET_ADJUST | 0x00 | Drive mode: Offset removal |
| | MTR_MODE_DRIVE | 0x01 | Drive mode: Normal drive |
| | MTR_ERROR_NONE | 0x0000 | No error |
| | MTR_ERROR_OVER_CURRENT_HW | 0x0001 | Hardware overcurrent error |
| | MTR_ERROR_OVER_VOLTAGE | 0x0002 | Overvoltage error |
| | MTR_ERROR_OVER_SPEED | 0x0004 | Overspeed error |
| | MTR_ERROR_UNDER_VOLTAGE | 0x0080 | Low-voltage error |
| | MTR_ERROR_OVER_CURRENT_SW | 0x0100 | Software overcurrent error |
| | MTR_ERROR_OVER_TEMPERATURE | 0x0200 | Overheat error |
| | MTR_ERROR_RESOLVER_DISCONNECT | 0x0400 | Resolver sensor disconnection error |
| | MTR_ERROR_RDC | 0x0800 | RDC error |
| | MTR_ERROR_UNKNOWN | 0xffff | Unknown error |
| | MTR_PWM_PERIOD_US | 1000.0f/MTR_CARRIER_FREQ | PWM update cycle |
| | MTR_DUTY_CYCLE_DEADTIME | MTR_DEADTIME/ MTR_PWM_PERIOD_US | Dead time duty value |

**Table 3.49 Macro Definitions in “r_mtr_foc_stm_rslv_control.h” (2/2)**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_foc_stm_rslv_control.h | MTR_MAX_DUTY_CYCLE | 1.0f-( MTR_DUTY_CYCLE_DEADTIME *0.5f +(MTR_AD_SAMPLING_TIME/ MTR_PWM_PERIOD_US)) | Maximum duty value |
| | MTR_MIN_DUTY_CYCLE | MTR_DUTY_CYCLE_DEADTIME *0.5f | Minimum duty value |

**Table 3.50 Macro Definitions in "r_mtr_parameter.h" (1/2)**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_parameter.h | MTR_M | MP_MAGNETIC_FLUX | Magnetic flux (Wb) |
| | MTR_R | MP_RESISTANCE | Resistance (Ω) |
| | MTR_LD | MP_D_INDUCTANCE | d-axis inductance (H) |
| | MTR_LQ | MP_Q_INDUCTANCE | q-axis inductance (H) |
| | MTR_J | MP_ROTOR_INERTIA | Inertia (kgm^2) |
| | MTR_NOMINAL_CURRENT_RMS | MP_NOMINAL_CURRENT_RMS | Rated current (Arms) |
| | MTR_POLE_PAIRS | MP_POLE_PAIRS | Number of pole pairs |
| | MTR_RPM_RAD | (MTR_POLE_PAIRS * MTR_TWOPI) / 60.0f | Coefficient of conversion from rpm to rad/s |
| | MTR_TWOPI_INV | 1.0f/MTR_TWOPI | Reciprocal of 2π |
| | MTR_RSLV_TIMER_CLOCK | MTR_PWM_TIMER_FREQ * 1000000 | Excitation frequency |
| | MTR_RSLV_TIMER_PERIOD | (float)(1/MTR_RSLV_TIMER_CLOCK) | Excitation frequency count value |
| | MTR_RSLV_POLE | RESOLVER_POLE | Number of poles of resolver sensor |
| | MTR_RSLV_POLE_INV | 1.0f/MTR_RSLV_POLE | Reciprocal of number of poles |
| | MTR_RSLV_ONE_CYCLE_RAD_MEC | MTR_TWOPI/MTR_RSLV_POLE | One-cycle angle of resolver sensor |
| | MTR_RSLV_EXCITATION_FREQ | EXCITATION_SIGNAL_FREQUENCY | Excitation frequency of resolver sensor |
| | MTR_RSLV_EXCITATION_CNT | MTR_RSLV_TIMER_CLOCK / MTR_RSLV_EXCITATION_FREQ | Excitation frequency count value of resolver sensor |
| | MTR_RSLV_EXCITATION_CNT_INV | 1.0f/MTR_RSLV_EXCITATION_CNT | Reciprocal of one-cycle count value of resolver sensor |
| | MTR_RSLV_EXCITATION_HALF_CNT | MTR_RSLV_EXCITATION_CNT/2.0f | 1/2-cycle count value of resolver sensor |
| | MTR_RSLV_ANGLE_DIFF | MTR_RSLV_ONE_CYCLE_RAD_MEC / MTR_RSLV_EXCITATION_CNT | One-count electrical angle of resolver sensor |
| | MTR_ANGLE_ADJ_TIME | 512 | Resolver position offset detection time |
| | MTR_RSLV_ANGLE_1PULSE_RAD | (float)(MTR_TWOPI/ MTR_RSLV_EXCITATION_CNT) | One-count mechanical angle of position signal (rad) |
| | MTR_PULSE_PER_REV | EP_PULSE_PER_REV | Number of pulses per rotation (PPR) |
| | MTR_RESO_MULTIPL | EP_RESO_MULTIPL | Multiplication resolution setting |
| | MTR_CURRENT_OMEGA | CP_CURRENT_OMEGA | Natural frequency of current control loop (Hz) |
| | MTR_CURRENT_ZETA | CP_CURRENT_ZETA | Attenuation coefficient for current control loop |
| | MTR_SPEED_OMEGA | CP_SPEED_OMEGA | Natural frequency of speed control loop (Hz) |
| | MTR_SPEED_ZETA | CP_SPEED_ZETA | Attenuation coefficient for speed control loop |
| | MTR_SPEED_LPF_OMEGA | CP_SPEED_LPF_OMEGA | Natural frequency of speed LPF (Hz) |
| | MTR_POS_OMEGA | CP_POS_OMEGA | Natural frequency of position control loop (Hz) |
| | MTR_FREQ_BAND_LIMIT | 3.0f | Band limit between control loops (times) |
| | MTR_MAX_CURRENT_OMEGA | 1500.0f | Maximum natural frequency of current control loop (Hz) |
| | MTR_MIN_POS_OMEGA | 1.0f | Minimum natural frequency of position control loop (Hz) |

**Table 3.50 Macro Definitions in "r_mtr_parameter.h" (2/2)**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_parameter.h | MTR_DEADTIME | IP_DEADTIME | Dead time (μs) |
| | MTR_CURRENT_RANGE | IP_CURRENT_RANGE | A/D conversion range for current (A) (p-p value) |
| | MTR_VDC_RANGE | IP_VDC_RANGE | A/D conversion range for inverter bus voltage (V) |
| | MTR_OFFSET_CALC_TIME | 256.0f | Current offset calculation time |
| | MTR_OFFSET_CALC_WAIT | (uint16_t)(MTR_LD/MTR_R*1000.0f* 10.0f*1.5f) | Offset calculation wait time |
| | MTR_PERIOD_MAG_VALUE | 1.5f | Control cycle delay time coefficient |
| | MTR_I_LIMIT_VD | IP_INPUT_V | Current PI output limit value |
| | MTR_I_LIMIT_VQ | IP_INPUT_V | Current PI output limit value |
| | MTR_REF_ID | CP_OL_ID_REF | d-axis current reference value |
| | MTR_I_LIMIT_IQ | MTR_NOMINAL_CURRENT_RMS * MTR_SQRT_2 | Speed PI output limit value |
| | MTR_LIMIT_IQ | MTR_NOMINAL_CURRENT_RMS * MTR_SQRT_2 | q-axis current limit value |
| | MTR_OVERCURRENT_ MARGIN_MULT | 1.5f | Overcurrent limit value coefficient |
| | MTR_OVERCURRENT_LIMIT | MTR_NOMINAL_CURRENT_RMS * MTR_SQRT_2 * MTR_OVERCURRENT_MARGIN_ MULT | Overcurrent limit value |
| | MTR_OVERVOLTAGE_LIMIT | IP_OVERVOLTAGE_LIMIT | Overvoltage limit value |
| | MTR_UNDERVOLTAGE_LIMIT | IP_UNDERVOLTAGE_LIMIT | Low-voltage limit value |
| | MTR_INPUT_V | IP_INPUT_V | Input voltage |
| | MTR_MCU_ON_V | MTR_INPUT_V * 0.8f | System power ON voltage |
| | MTR_RATE_LIMIT_CURRENT | 0.0005f | Current increase rate limit value |
| | MTR_SPEED_CTRL_PERIOD | 0.00025f | Current control cycle |
| | MTR_SPEED_FF_RATIO | 0.6f | Speed feedforward coefficient |
| | MTR_MIN_SPEED_RPM | CP_MIN_SPEED_RPM | Minimum speed (rpm) |
| | MTR_MAX_SPEED_RPM | CP_MAX_SPEED_RPM | Maximum speed (rpm) |
| | MTR_MAX_SPEED_RAD | MTR_MAX_SPEED_RPM * MTR_TWOPI_60 | Maximum speed (rad/s) |
| | MTR_SPEED_LIMIT_RPM | CP_SPEED_LIMIT_RPM | Speed limit value (rpm) |
| | MTR_OVERSPEED_LIMIT_ RAD | MTR_SPEED_LIMIT_RPM * MTR_RPM_RAD | Speed limit value (rad/s) |
| | MTR_RATE_LIMIT_SPEED | 0.3f | Acceleration rate limit value |
| | MTR_POS_DEAD_BAND | 1.0f | Position dead band (±count) |
| | MTR_POS_BAND_LIMIT | 3.0f | Positioning completed flag range (±count) |
| | MTR_POS_INTERVAL_TIME | 400 | Position reference update interval time |
| | MTR_POS_LIMIT | (32767.0f * MTR_TWOPI) / 360.0f | Position reference limit value |
| | MTR_OFFSET_ALGIN_ID_ RATIO | (0.8f) | The coefficient to determine d-axis current command, when removing position sensor offset* |

Note: * d-axis current command = Motor norminal current[Arms] * $\sqrt{2}$ * MTR_OFFSET_ALIGN_ID_RATIO

**Table 3.51 Macro Definitions in "r_mtr_common.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_common.h | MTR_TWOPI | 2.0f * 3.1415926535f | 2π |
| | MTR_TWOPI_60 | MTR_TWOPI / 60.0f | 2π / 60 |
| | MTR_DIG_RAD | MTR_TWOPI / 360.0f | 2π / 360 |
| | MTR_SQRT_2 | 1.41421356f | √2 |
| | MTR_SQRT_3 | 1.7320508f | √3 |
| | MTR_CW | 0 | Forward rotation |
| | MTR_CCW | 1 | Reverse rotation |
| | MTR_LED_ON | 0 | LED turned on |
| | MTR_LED_OFF | 1 | LED turned off |
| | MTR_FLG_CLR | 0 | Flag cleared |
| | MTR_FLG_SET | 1 | Flag set |
| | MTR_ENABLE | 1 | Enabled |
| | MTR_DISABLE | 0 | Disabled |
| | MTR_ID_A | 0 | Motor ID A |
| | MTR_ID_B | 1 | Motor ID B |
| | TRUE | 1 | True |
| | FALSE | 0 | False |
| | ON | 1 | |
| | OFF | 0 | |
| | HIGH | 1 | |
| | LOW | 0 | |
| | OK | 0x00 | |
| | NG | 0xFF | |

**Table 3.52 Macro Definitions in "r_mtr_position_profiling.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_position_profiling.h | MTR_MAX_ACCEL | (MTR_POLE_PAIRS * MTR_NOMINAL_CURRENT_RMS * MTR_SQRT_2 * MTR_M) / MTR_J | Maximum acceleration |
| | MTR_ACCEL_TIME | 0.06f | Acceleration time of reference speed (s) |
| | MTR_MAX_ACCEL_TIME | MTR_MAX_SPEED_RAD / MTR_MAX_ACCEL | Maximum acceleration time of reference speed (s) |
| | MTR_TIME_SEC | MTR_SPEED_CTRL_PERIOD | 250 (μs) |
| | MTR_CTRL_TRIANGLE | 0 | |
| | MTR_CTRL_TRAPEZOIDAL | 1 | |
| | MTR_POS_STEADY_STATE | 0 | |
| | MTR_POS_TRANSITION_STATE | 1 | |

### 3.12.4 “r_mtr_spm_rslv_foc_rx/src/functions” Folder

**Table 3.53 Macro Definitions in “r_mtr_volt_err_comp.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_volt_err_comp.h | MTR_VOLT_COMP_LIMIT | MTR_CARRIER_FREQ * MTR_DEADTIME * 2.0f) / 1000.0f | Coefficient for calculating voltage compensation limit |
| | ARY_SIZE | 5 | Number of data items |
| | TBL_COMP_V0 | 0.205312148f | Voltage compensation value (V) |
| | TBL_COMP_V1 | 0.335981607f | |
| | TBL_COMP_V2 | 0.414872766f | |
| | TBL_COMP_V3 | 0.457023382f | |
| | TBL_COMP_V4 | 0.46840322f | |
| | TBL_COMP_I0 | 0.038731616f | Current (A) |
| | TBL_COMP_I1 | 0.0816742256f | |
| | TBL_COMP_I2 | 0.140160277f | |
| | TBL_COMP_I3 | 0.193200409f | |
| | TBL_COMP_I4 | 0.217510894f | |
| | MTR_VOLT_ERR_COMP_ENABLE | 1 | Voltage error compensation enabled |
| | REF_VOLTAGE | 24.0f | Power-supply voltage (V) |

**Table 3.54 Macro Definitions in “r_mtr_speed_observer.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_speed_observer.h | SOB_SPEED_CTRL_PERIOD | MTR_CTRL_PERIOD | Control period (speed loop) (s) |
| | SOB_OMEGA | CP_SOB_OMEGA | Natural frequency of speed observer |
| | SOB_ZETA | CP_SOB_ZETA | Damping ratio for speed observer |
| | SOB_HPF_OMEGA | 1.0f | Natural frequency of speed observer HPF |
| | SOB_HPF_K2 | 2.0f / SOB_SPEED_CTRL_PERIOD | Speed observer HPF gain |
| | SOB_HPF_K3 | -(1.0f - SOB_HPF_K2) | Speed observer HPF gain |

**Table 3.55 Macro Definitions in “r_mtr_mod.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_mod.h | MOD_DEFAULT_MAX_DUTY | 1.0f | Default maximum duty cycle |
| | MOD_METHOD_SPWM | 0 | Sinusoidal pulse-width-modulation |
| | MOD_METHOD_SVPWM | 1 | Space vector pulse-width-modulation |
| | MOD_SATFLAG_BITU | 1 << 0 | Saturation flag bit for U phase |
| | MOD_SATFLAG_BITV | 1 << 1 | Saturation flag bit for V phase |
| | MOD_SATFLAG_BITW | 1 << 2 | Saturation flag bit for W phase |
| | MOD_VDC_TO_VAMAX_MULT | 0.97f | Basic coefficient used to convert Vdc to Vamax 1:1 |
| | MOD_SVPWM_MULT | 1.155f | The usable voltage is multiplied by sqrt (4/3) when using SVPWM. |

**Table 3.56 Macro Definitions in "r_mtr_fluxwkn.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_fluxwkn.h | FLUXWKN_DEF_VFWRATIO | 0.95f | Default flux-weakening voltage ratio |
| | FLUXWKN_DEF_VFWRATIO_MIN | 0.5f | Default minimum flux-weakening voltage ratio |
| | FLUXWKN_STATE_BYPASSED | 0x0000 | Flux-weakening control is not necessary; the module is bypassed. |
| | FLUXWKN_STATE_FLUXWKN | 0x0001 | Normal flux-weakening state |
| | FLUXWKN_STATE_IDSAT | 0x0002 | d-axis current has already saturated. |
| | FLUXWKN_STATE_ERROR | 0x8000 | Flux-weakening general or runtime error |
| | FLUXWKN_STATE_INVALID_MOTOR | 0x9001 | Invalid motor parameter<br>(The specified motor pointer is null, or any of Ld, Lq, and flux is smaller than 0.) |
| | FLUXWKN_STATE_INVALID_IAMAX | 0x9002 | Invalid maximum Ia value (Iamax < 0.0) |
| | FLUXWKN_STATE_INVALID_VAMAX | 0x9003 | Invalid maximum Va value (Vamax < 0.0) |
| | FLUXWKN_STATE_INVALID_VFWRATIO | 0x9004 | Invalid maximum voltage ratio for flux-weakening |

**Table 3.57 Macro Definitions in "r_mtr_ipd.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_ipd.h | POS_LPF_OMEGA | 200.0f | Natural frequency of position HPF |
| | POS_LPF_ZETA | 1.0f | Damping ratio for position HPF |
| | POS_SPEED_RATIO | 2.0f | Speed gain ratio for IPD |
| | POS_FF_RATIO | 0.2f | Position feedforward ratio |
| | POS_KP_RATIO | 0.3f | Position KP ratio |
| | POS_IPD_ERR_LIMIT_1 | 10.0f | Position error limit 1 (rad) |
| | POS_IPD_ERR_LIMIT_2 | 0.2f | Position error limit 2 (rad/s) |
| | LPF_FLAG | LPF_ON | Initial set position LPF enabled or disabled |
| | LPF_OFF | 0 | Position LPF disabled |
| | LPF_ON | 1 | Position LPF enabled |

**Table 3.58 Macro Definitions in "r_rslv_compensation.h"**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_rslv_compensation.h | BPF_COMP_GAIN_MULT_CW | 1.00f | Multiplier for BPF compensation in clockwise rotation |
| | BPF_COMP_GAIN_MULT_CCW | 1.00f | Multiplier for BPF compensation in counterclockwise rotation |

### 3.12.5 “r_mtr_spm_rslv_foc_rx/src/mode” Folder

**Table 3.59 Macro Definitions in “r_mtr_statemachine.h”**

| File Name | Macro Name | Defined Value | Remark |
|---|---|---|---|
| r_mtr_statemachine.h | MTR_MODE_INACTIVE | 0x00 | |
| | MTR_MODE_ACTIVE | 0x01 | |
| | MTR_MODE_ERROR | 0x02 | |
| | MTR_SIZE_STATE | 3 | Number of states |
| | MTR_EVENT_INACTIVE | 0x00 | |
| | MTR_EVENT_ACTIVE | 0x01 | |
| | MTR_EVENT_ERROR | 0x02 | |
| | MTR_EVENT_RESET | 0x03 | |
| | MTR_SIZE_EVENT | 4 | Number of events |
| | MTR_STATEMACHINE_ERROR_NONE | 0x00 | No error flag |
| | MTR_STATEMACHINE_ERROR_EVENTOUTBOUND | 0x01 | The event index is out of bound. |
| | MTR_STATEMACHINE_ERROR_STATEOUTBOUND | 0x02 | The state index is out of bound. |
| | MTR_STATEMACHINE_ERROR_ACTIONEXCEPTION | 0x03 | The action function returns a failure. |

## 3.13 List of Variables for Analyzer Functions

Table 3.60 list input variables to be used for the analyzer user interface. The values specified in these variables are reflected in the corresponding variables in the middleware layer to be used for motor control when a value that is the same as the “g_u1_enable_write” value is written to “com_u1_enable_write”. However, the variables marked with an asterisk (*) in the tables are independent of “com_u1_enable_write”.

**Table 3.60 Input Variables for Analyzer Functions (1/2)**

| Variable Name for Analyzer | Type | Description |
|---|---|---|
| com_u1_mode_system (*) | uint8_t | State management 0: Stop mode 1: Run mode 3: Reset |
| com_u1_memory_write (*) | uint8_t | Writing data to flash memory 1: Enabled 0: Disabled |
| com_u1_flg_csig (*) | uint8_t | Carrier correction 0: Enabled 1: Disabled (default) |
| com_u1_flg_upd_csigparam (*) | uint8_t | Carrier correction value update 0: Stop 1: Start |
| com_u2_csig_shiftnum (*) | uint16_t | Carrier correction shift amount |
| com_u2_csig_amplvl (*) | uint16_t | Carrier correction gain |
| com_u1_flg_rdc_sequence (*) | uint8_t | RDC startup sequence execution flag 0: Stop 1: Start (default) |
| com_u1_enable_write | uint8_t | Variable rewriting enable (toggle operation) |
| com_u1_flag_angle_spl_comp_use | uint8_t | Sampling delay compensation of resolver position information 0: Disabled, 1: Enabled (default) |
| com_u1_flag_cc_spl_comp_use | uint8_t | Angle correction at the time of voltage reflection 0: Disabled, 1: Enabled (default) |
| com_u1_flag_bpf_delay_comp_use | uint8_t | Delay correction using bandpass filter of RDC 0: Disabled, 1: Enabled (default) |
| com_u1_flag_sob_use | uint8_t | Speed observer enable 0: Disabled, 1: Enabled (default) |
| com_u1_flag_volt_err_comp_use | uint8_t | Voltage error compensation enable 0: Disabled, 1: Enabled (default) |
| com_u1_flag_flux_weakening_use | uint8_t | Flux-weakening enable 0: Disabled, 1: Enabled (default) |
| com_u1_ctrl_loop_mode | uint8_t | Control loop switching<br>0: d-axis current control<br>1: q-axis current control<br>2: Speed control (default)<br>3: Position control |
| com_u1_ctrl_method_mode | uint8_t | Control method switching<br>0: PID control (position P, speed PI, current PI) (default)<br>1: IPD control (position and speed IPD + position FF + speed FF + position P and current PI)<br>FF: Feedforward control |
| com_u1_position_input_mode | uint8_t | Position reference value input method switching<br>0: Zero reference<br>1: Direct input (stepped input)<br>2: Creating trapezoidal speed reference (default) |
| com_u1_offset_adjust_mode | uint8_t | Magnetic pole position detection mode<br>0: Current (default)<br>1: Position<br>2: Offset removal completed |
| com_u1_direction | uint8_t | Direction of rotation 0: Clockwise 1: Counterclockwise |
| com_f4_ref_position_deg | int16_t | Position reference value in mechanical angle (degree) |
| com_s2_ref_speed_rpm | int16_t | Speed reference value in mechanical angle (rpm) |
| com_u2_min_speed_rpm | uint16_t | Minimum speed value in mechanical angle (rpm) |
| com_u2_max_speed_rpm | uint16_t | Maximum speed value in mechanical angle (rpm) |
| com_u2_speed_limit_rpm | uint16_t | Speed limit value in mechanical angle (rpm) |
| com_u2_pos_interval_time | uint16_t | Normal wait time for position response |
| com_u2_pos_deadband | uint16_t | Dead band |
| com_u2_pos_band_limit | uint16_t | Position error zero range |
| com_u2_encd_cpr_mech | uint16_t | Number of encoder pulses |
| com_u2_offset_calc_time | uint16_t | Current offset value calculation time (ms) |
| com_u2_offset_calc_wait | uint16_t | Current offset calculation wait time (ms) |
| com_u2_run_mode | uint16_t | RUN mode 0: Offset removal 1: Normal drive |
| com_f4_ipd_speed_k_ratio | float | Magnification of speed gain for IPD control |
| com_f4_ipd_pos_kp_ratio | float | Magnification of position P control amount for IPD control |
| com_f4_ipd_err_limit_1 | float | IPD control difference limit 1 |
| com_f4_ipd_err_limit_2 | float | IPD control difference limit 2 |
| com_f4_accel_time | float | Acceleration time (s) for creating a position reference value |
| com_f4_speed_rate_limit | float | Acceleration rate limit |

**Table 3.60 Input Variables for Analyzer Functions (2/2)**

| Variable Name for Analyzer Functions | Type | Description |
|---|---|---|
| com_f4_ref_id | float | d-axis current reference value |
| com_f4_ref_iq | float | q-axis current reference value |
| com_f4_current_rate_limit | float | Current increase rate limit |
| com_f4_bpf_comp_gain_cw | float | Adjustment coefficient for resolver BPF compensation (clockwise) |
| com_f4_bpf_comp_gain_ccw | float | Adjustment coefficient for resolver BPF compensation (counterclockwise) |
| com_f4_bpf_comp_base_cw | float | Base coefficient for resolver BPF compensation (clockwise) |
| com_f4_bpf_comp_base_ccw | float | Base coefficient for resolver BPF compensation (counterclockwise) |
| com_u2_mtr_pp | uint16_t | Number of pole pairs |
| 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 | Magnetic flux (Wb) |
| com_f4_mtr_j | float | Rotor inertia (kgm^2) |
| com_f4_nominal_current_rms | float | Rated current (Arms) |
| com_f4_current_omega | float | Natural frequency of current control loop |
| com_f4_current_zeta | float | Attenuation coefficient for current control loop |
| com_f4_speed_omega | float | Natural frequency of speed control loop |
| com_f4_speed_zeta | float | Attenuation coefficient for speed control loop |
| com_f4_pos_omega | float | Natural frequency of position control loop |
| com_f4_sob_omega | float | Natural frequency of speed observer |
| com_f4_sob_zeta | float | Attenuation coefficient for speed observer |
| com_f4_id_kp | float | Proportional gain for d-axis current PI control |
| com_f4_id_ki | float | Integral gain for d-axis current PI control |
| com_f4_iq_kp | float | Proportional gain for q-axis current PI control |
| com_f4_iq_ki | float | Integral gain for q-axis current PI control |
| com_f4_speed_kp | float | Proportional gain for speed PI control |
| com_f4_speed_ki | float | Integral gain for speed PI control |
| com_f4_pos_kp | float | Proportional gain for position P control (PID control mode)<br>Proportional gain for IPD control + proportional gain for position P control (IPD control mode) |
| com_u1_rslv_calibmode | uint8_t | Automatic calibration by the RDC<br>1: Adjustment of the gain and phase of the resolver signals<br>2: Adjustment of the angle error correction signal |

# 4. Basic Operating Procedure

## 4.1 Motor Control Development Support Tool “Renesas Motor Workbench”

The software covered by this application note uses the motor control development support tool “Renesas Motor Workbench” as a user interface (rotation and stop reference setting, rotational speed reference setting, etc.). For details about usage of this tool, see *Renesas Motor Workbench 2.0 User’s Manual*.

Obtain the motor control development support tool “Renesas Motor Workbench” from the Renesas Electronics website.

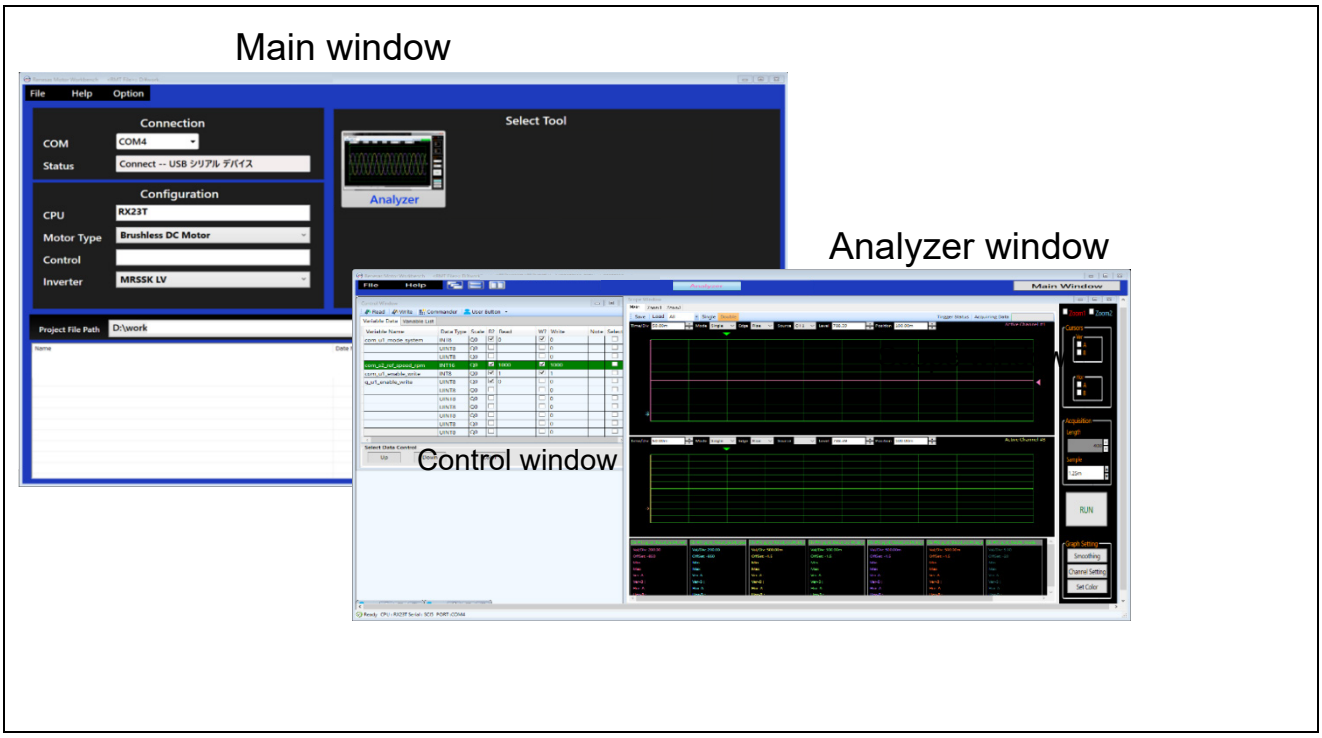
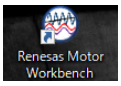



**Figure 4.1 Windows of Renesas Motor Workbench**

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

1. Click the tool icon 

to activate the tool.
2. From the menu bar on the Main panel, select [RMTFile] → [Open RMT File(O)].
   Read the RMT file in the “ics” folder in the project folder.
3. Select the COM port of the connected kit in the [COM] box under “Connection”.
4. Click the [Analyzer] button in the “Select Tool” window to display the analyzer function window.
   When connections have been properly made, a message “Ready CPU: RX24TSerial: SCI5 PORT: COM(X)” appears at the lower left of the window.


5. Activate the motor according to section 4.2, Example of Motor Driving Operation Using Analyzer Functions.

## 4.2 Example of Motor Driving Operation Using Analyzer Functions

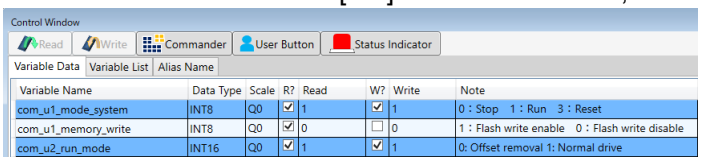
This section describes how to drive the motor using the analyzer functions of the RMW.

(1) When motor current or resolver position offset has not been removed

The following describes operation when the motor current or resolver position offset has not been removed. When the offset has been removed and the motor current and resolver option offset have been written to the flash memory, this step can be skipped.

Executing offset removal

1. Write 0 to the com_u2_run_mode variable to enter offset removal mode.
   Confirm that the checkbox in the [W?] column is selected, and then write 0 to the [Write] column.




| Variable Name | Data Type | Scale | R? | Read | W? | Write | Note |
|---|---|---|---|---|---|---|---|
| com_u1_mode_system | INT8 | Q0 | ☑ | 1 | ☑ | 1 | 0：Stop 1：Run 3：Reset |
| com_u1_memory_write | INT8 | Q0 | ☑ | 0 | ☐ | 0 | 1：Flash write enable 0：Flash write disable |
| com_u2_run_mode | INT16 | Q0 | ☑ | 1 | ☑ | 1 | 0: Offset removal 1: Normal drive |

2. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| com_u1_enable_write | INT8 | Q0 | ☑ | 1 | ☑ | 0 | Allow variable rewriting (toggle operation) |
| g_u1_enable_write | UINT8 | Q0 | ☑ | 0 | ☐ | 0 | ↑ Rewrite reference value |

   Click the [Write] button.
3. Write 1 to com_u1_mode_system and then click the [Write] button to execute offset removal. Upon completion of offset removal, the com_u1_mode_system value automatically becomes 0.
   The measurement result is reflected in each variable as shown below.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| g_st_foc.f4_rslv_angle_offset_rad | FLOAT | Q0 | ☑ | 0.3600244 | ☐ | 0 | Resolver angle offset [rad] |
| g_st_foc.f4_offset_ia | FLOAT | Q0 | ☑ | -0.04605473 | ☐ | 0 | α-axis current offset [A] |
| g_st_foc.f4_offset_ib | FLOAT | Q0 | ☑ | -0.03246146 | ☐ | 0 | β-axis current offset [A] |

   Variable values are removed as offset when the motor is driven.

Writing to flash memory

1. Write 1 to com_u1_memory_write and then click the [Write] button. Each offset value is written to the flash memory and the written values are retained even after power is turned on again. Upon completion of writing, the com_u1_memory_write value automatically becomes 0.

(2) Driving in speed control mode

Entering speed control mode

1. Write 1 to the com_u2_run_mode variable to enter normal drive mode.
2. Write 2 to the com_u1_ctrl_loop_mode variable to enter speed control mode.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| com_u1_enable_write | INT8 | Q0 | ☑ | 1 | ☑ | 0 | Allow variable rewriting (toggle operation) |
| g_u1_enable_write | UINT8 | Q0 | ☑ | 0 | ☐ | 0 | ↑ Rewrite reference value |

   Click the [Write] button.

Setting the speed reference value to drive the motor

1. Specify the speed reference value (rpm) in com_s2_ref_speed_rpm.
2. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| com_u1_enable_write | INT8 | Q0 | ☑ | 1 | ☑ | 0 | Allow variable rewriting (toggle operation) |
| g_u1_enable_write | UINT8 | Q0 | ☑ | 0 | ☐ | 0 | ↑ Rewrite reference value |

   Click the [Write] button.
3. Write 1 to the com_u1_mode_system variable and then click the [Write] button to enter Run mode.
   The motor starts rotating.

(3) Driving in position control mode

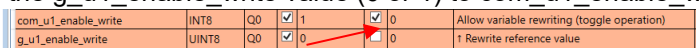
Entering position control mode

1. Write 1 to the com_u2_run_mode variable to enter normal drive mode.
2. Write 3 to the com_u1_ctrl_loop_mode variable to enter position control mode.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.


   Click the [Write] button.

Setting position reference value to drive the motor

1. Write 1 to the com_u1_mode_system variable and then click the [Write] button to enter Run mode.
   The motor starts the positioning control operation.
2. Specify the target position (degree) in com_f4_ref_position_deg.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.




   Click the [Write] button to execute positioning to follow the target value.

(4) Recovering from an error

Occurrence of an error

When an error occurs, the motor stops working and the error status is reflected in g_st_foc.u2_error_status.

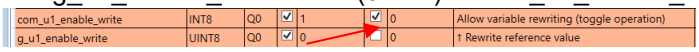

Clearing the error

1. Write 3 to the com_u1_mode_system variable and then click the [Write] button to enter reset mode.
   The error status is cleared and all variables are reset.
   The g_st_foc.u2_error_status variable is cleared to 0.

(5) Open-loop driving without using sensors (with current control)

Entering d-axis current control mode

1. Write 1 to the com_u2_run_mode variable to enter normal drive mode.
2. Write 0 to the com_u1_ctrl_loop_mode variable to enter d-axis current control mode.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.




   Click the [Write] button.

Setting speed and d-axis current reference values to drive the motor

1. Specify the speed reference value (rpm) in com_s2_ref_speed_rpm.
2. Specify the current reference value in com_f4_ref_id.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.


   Click the [Write] button.
4. Write 1 to the com_u1_mode_system variable and then click the [Write] button to enter Run mode.
   The motor starts rotating.

(6) q-axis current control

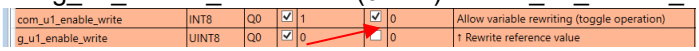
Entering q-axis current control mode

1. Write 1 to the com_u2_run_mode variable to enable normal drive mode.
2. Write 1 to the com_u1_ctrl_loop_mode variable to enable q-axis current control mode.
3. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.


   Click the [Write] button.

Setting q-axis current reference value to drive the motor

1. Specify the current reference value in com_f4_ref_iq.
2. Rewrite (toggle) the com_u1_enable_write variable and reflect the variable value.
   Write the g_u1_enable_write value (0 or 1) to com_u1_enable_write.

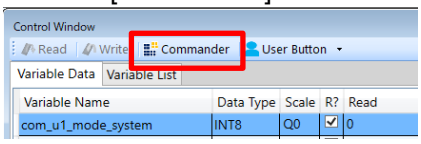



   Click the [Write] button.
3. Write 1 to the com_u1_mode_system variable and then click the [Write] button to enter Run mode.
   The motor starts rotating.

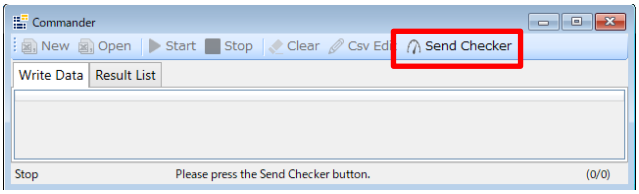
(1) Position control using the commander

Activating the commander

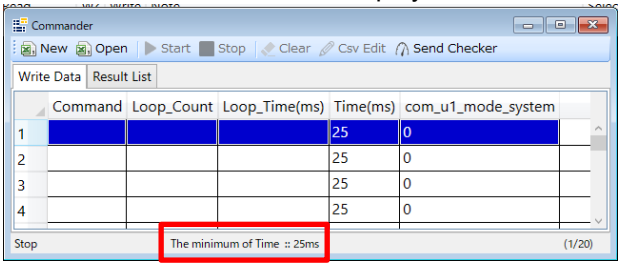
1. Click the [Commander] button in the Control window.



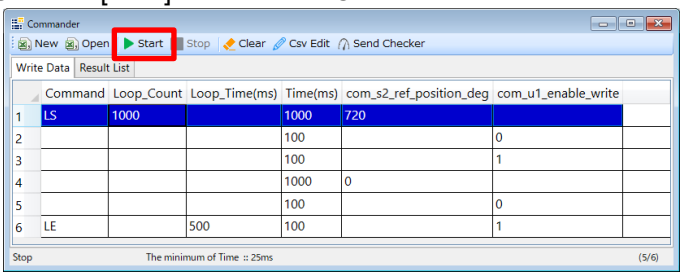

2. The Commander window appears. On this window, click the [Send Checker] button to check the data transmission rate.




The minimum transfer time is displayed.




3. Click the [Open] button to read the “Position_test.csv” file. Select position control mode, write 1 to the com_u1_mode_system variable, and then click the [Write] button to enter Run mode. The motor starts the positioning control operation.
4. Click the [Start] button in the Commander window to start the sequence operation.



## Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

## Revision History

| | | Description | |
|---|---|---|---|
| **Rev.** | **Date** | **Page** | **Summary** |
| 1.00 | Jan.08.20 | — | First edition issued |
| 1.10 | Feb.14.20 | 9 | Table 2.5 Folder and File Configuration<br>A file name was corrected. |
| | | 47 | Table 3.17 Current Control Processing Functions (2/3)<br>A file name was corrected. |
| | | 70 | Table 3.19 Speed Control Processing Functions<br>A file name was corrected. |
| | | 72 | Table 3.20 Speed Reference Value Creation Processing Functions<br>A file name was corrected. |
| | | 75 | Table 3.22 Flux-Weakening Control Processing Functions<br>A file name was corrected. |
| | | 82 | Table 3.26 Structure Definitions in “r_mtr_stm_rslv_foc_rx_if.h” (1/2)<br>A variable name was corrected. |
| | | 104 | Table 3.60 Input Variables for Analyzer Functions (1/2)<br>A variable name for analyzer was corrected. |
| | | 108 | 4.2, (3) Driving in position control mode<br>A variable name for analyzer was corrected. |
| 1.20 | Mar.16.20 | 5 | Figure 2.1 Hardware Configuration: Corrected |
| | | 6 | Table 2.1 User Interfaces: Corrected |
| | | 6 | Table 2.2 Pin Interfaces: Corrected |
| | | 9 | Table 2.5 Folder and File Configuration:<br>The description for file r_mtr_ipd.h, r_mtr_ipd.obj was corrected. |
| | | 42 | Section 3.9 Calibration of Errors: Added |
| | | 94 | Table 3.47 Macro Definitions in “r_resolver_rdc_command.h”:<br>RDC_CMD_INITIAL_VALUE_12h deleted |
| | | 105 | Table 3.60 Table 3.60 Input Variables for Analyzer Functions (2/2): com_u1_rslv_calibmode added |
| 1.30 | Mar.29.21 | 1 | Target Sample Program was corrected. |
| | | 4 | Updated version of CC-RX toolchain to 3.03.00<br>Updated version of CS+ to 8.05.00<br>Added version of e$^2$studio |
| | | 10 | Fixed description for auto_generation folder in<br>Table 2.5 Folder and File Configuration (2/2) |
| | | 12 | Table 2.6 Basic Specifications of the Vector Control Software:<br>ROM and RAM size was corrected. |
| | | 67 | Updated flowchart of position offset removing process function |
| | | 85 | Changed the name of strcuture menber s2_rslv_cycle_cnt to s4_rslv_cycle_cnt, changed the type from int16_t to int32_t on Table 3.27 Structure Definitions in “r_mtr_foc_stm_rslv_control.h” (2/3) |
| | | 99 | Added the description of new macro MTR_OFFSET_ALIGN_ID_RATION on Table 3.50 Macro Definitions in “r_mtr_parameter.h” (2/2) |
| 1.31 | Apr.23.21 | — | Fixed issue of flash memory operation.<br>Updated version to 1.31 |

# 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/.