# 120 degrees conducting control by Sensor-less drive

## RAJ306000 implementation guide

### Summary

This application note explains a sample program to support the Sensor-less 120-degrees conducting control of 3-Phase brushless DC motor using RAJ306000, and the method using the library of development support tool "In Circuit Scope".

These sample programs are only to be used as reference and Renesas Electronics Corporation does not guarantee the operations. Please use them after carrying out a thorough evaluation in a suitable environment.

### Operation checking device

Operations of the sample program are checked by using the following device.

- RAJ306000

### Target of sample program

A sample program that this application note is intended shown below.

- RAJ306000_LESS_120_OPEN_CSP_CA_V103 (IDE: CS+ for CA, CX)
- RAJ306000_LESS_120_OPEN_CSP_CC_V103 (IDE: CS+ for CC)
- RAJ306000_LESS_120_OPEN_E2S_CC_V103 (IDE: e$^2$ studio)

  120-degrees conducting control sample program with Sensor-less drive for RAJ306000 (Complementary PWM Mode)

### Reference materials

- RL78/G1F User's Manual: Hardware (R01UH0516EJ0110)
- RAJ306000 Series User's Manual: Hardware (R18UZ0066EJ0100)
- In Circuit Scope Manual

  Downloadable from: http://www.desktoplab.co.jp/download.html

## Contents

# 1. Overview

This application note explains a sample program to support the Sensor-less 120-degrees conducting control of 3-Phase brushless DC motor using RAJ306000, and the method using the library of development support tool "In Circuit Scope". (Note 1).

Note:

1. The development support tool In Circuit Scope (ICS) is a product of Desk Top Laboratories Inc.

   Desk Top Laboratories Inc. (http://www.desktoplab.co.jp/)

## 1.1 Development environment

Development environment of the sample programs are showed in Table 1-1 and Table 1-2.

**Table 1-1 Software development environment**

| | |
|---|---|
| **Integrated Development Environment** | CS+ for CA, CX V3.02.00 [15 Mar 2016] |
| **Compiler** | CA78K0R V1.72 |

| | |
|---|---|
| **Integrated Development Environment** | CS+ for CC V6.01.00 [01 Dec 2017] |
| **Compiler** | CC-RL V1.06.00 |

| | |
|---|---|
| **Integrated Development Environment** | $e^2$ studio Version: 5.4.0.015 |
| **Compiler (Toolchain)** | CC-RL V1.06.00 |

**Table 1-2 Hardware development environment**

| | |
|---|---|
| **On-chip Debugging Emulator** | RENESAS E1 Emulator (R0E000010KCE00) |
| **Operation Checking Device** | RAJ306000 (Note 2) |
| **RAJ306000 Series Evaluation Board** | RTK0EML2A0D00010BJ |

Note:

2. The configuration of RAJ306000 which is a SIP product containing MCU (RL78/G1F) and PreDriver is shown in Figure 1-1.

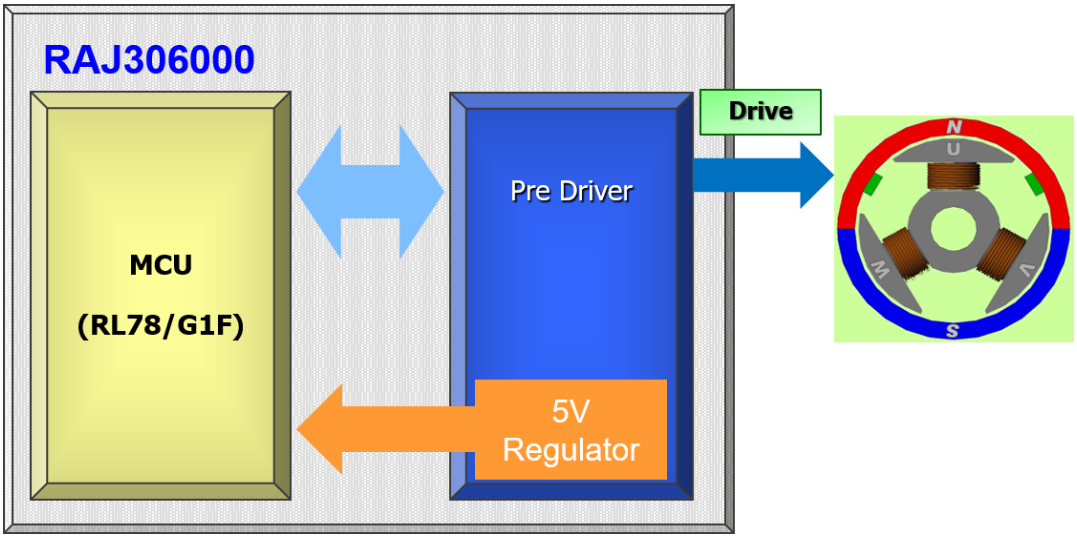


**Figure 1-1 RAJ306000**

## 2. System overview

Overview of RAJ306000 system is shown in Figure 2-1.

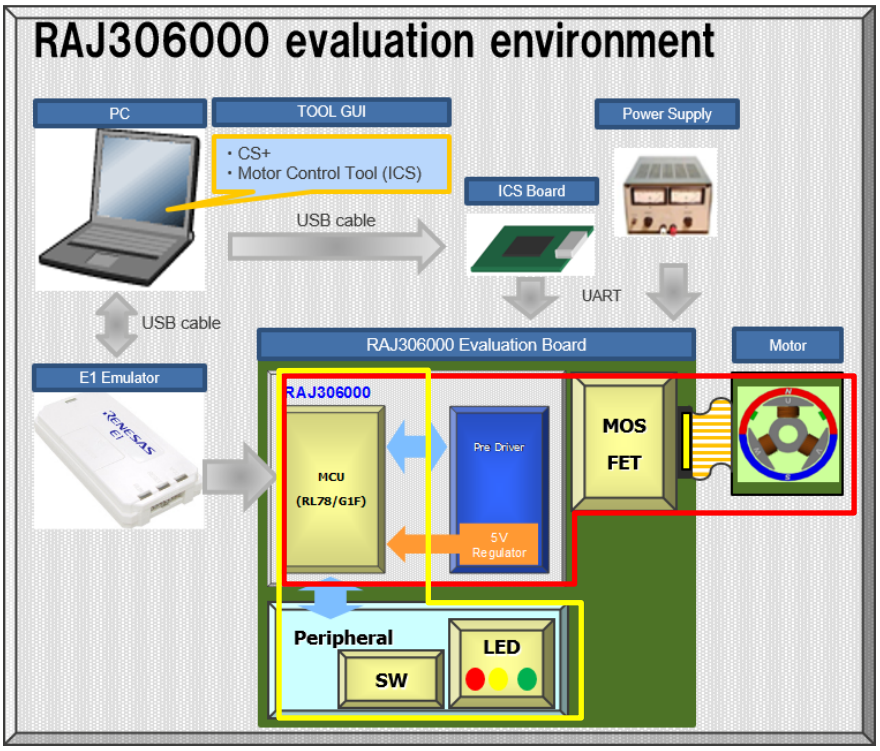


**Figure 2-1 System configuration**

## 2.1 Hardware configuration

Hardware configurations are shown below:

Figure 2-2 Hardware connection of between RL78/G1F and PreDriver.

Note: These are the hardware blocks highlighted in RED in Figure 2-1.

Figure 2-3 Hardware connection of between RL78/G1F and Peripheral.

Note: These are the hardware blocks highlighted in YELLOW in Figure 2-1.

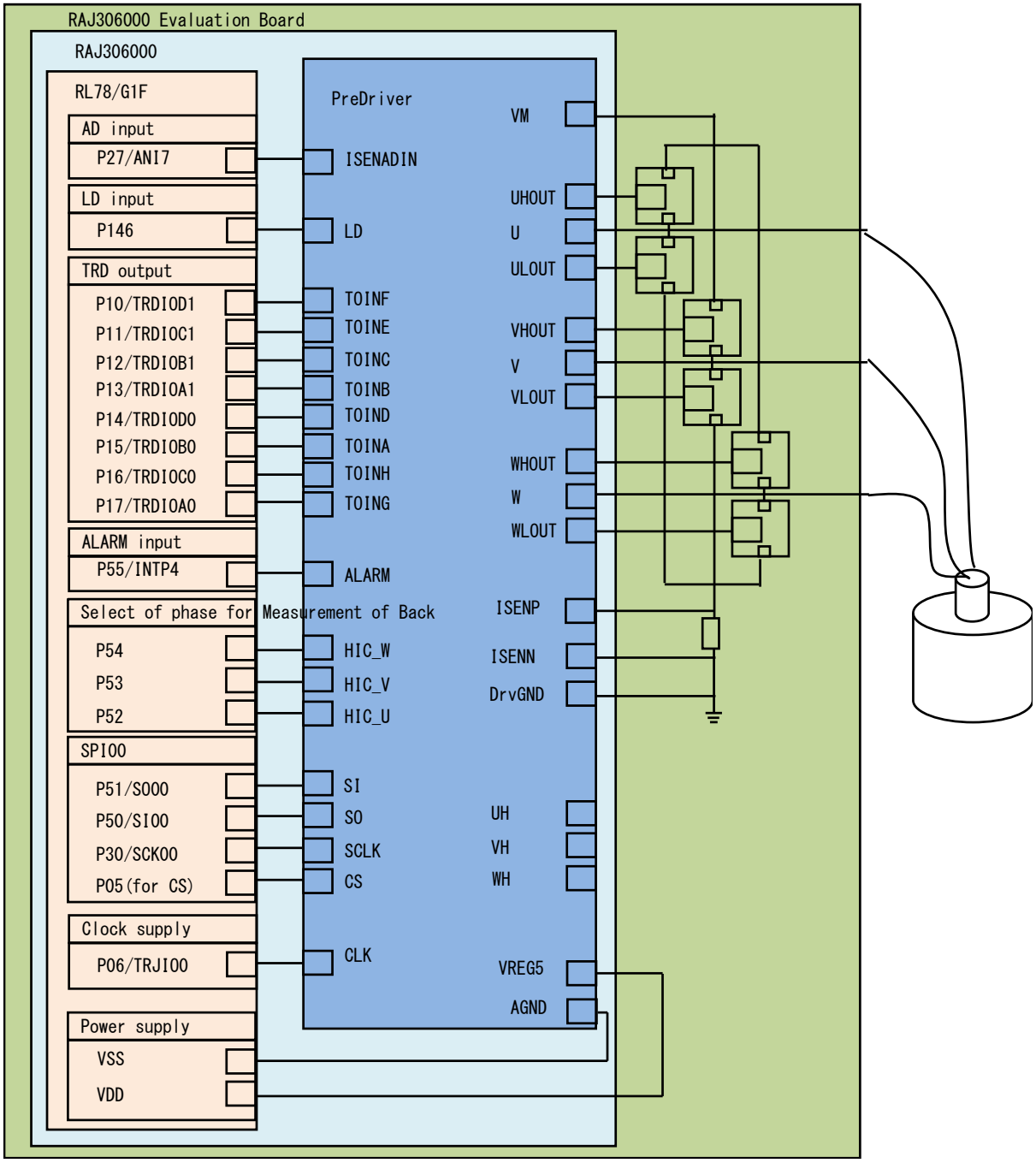


**Figure 2-2 Hardware Configuration Diagram (RL78/G1F, PreDriver)**

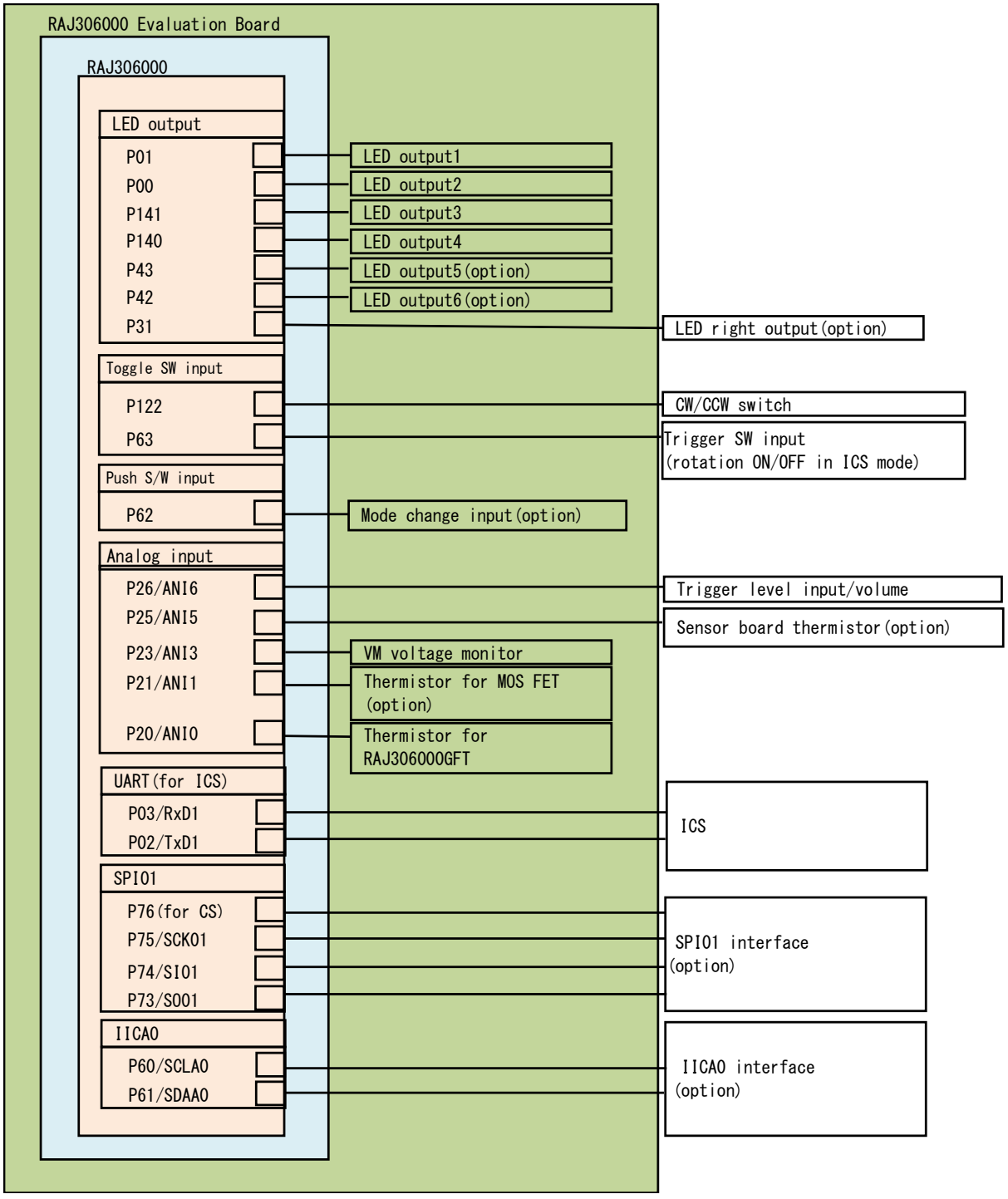


**Figure 2-3 Hardware Configuration Diagram (RL78/G1F, Peripheral)**

## 2.2 Hardware specifications

### 2.2.1 User interface

List of user interfaces of this system is shown in Table 2-1.

**Table 2-1 User Interface**

| Item | Interface component | Function |
|---|---|---|
| Rotation direction | Selector switch of CW/CCW (SW1) or ICS | Input of rotation direction (CW/CCW) |
| Rotation speed | Input of trigger level/volume (VR1) | Rotation speed command value input (analog value) |
| START/STOP | Input of trigger level/volume (VR1) or ICS | Motor rotation start/stop command |
| RED LED | LED output1 | ・At the time of normal operation: OFF<br>・At the time of error detection: ON |
| | LED output2 | ・At the time of stop: OFF<br>・At the time of Motor rotation: ON |
| | LED output3 | ・Rotation speed under 1500[rpm]: OFF<br>・Rotation speed over 1500[rpm]: ON |
| | LED output4 | ・Rotation speed under 3000[rpm]: OFF<br>・Rotation speed over 3000[rpm]: ON |
| | LED output5 | ・At the time of normal operation: OFF<br>・At the time of error detection: ON |
| | LED output6 | ・At the time of normal operation: OFF<br>・At the time of error detection: ON |
| Over voltage and under voltage detection | VM voltage detection | VM voltage measurement (input) |

List of interfaces of RL78/G1F micro controller of this system is shown in Table 2-2.

**Table 2-2 Port Interface (RL78/G1F)**

| Terminal name | Function |
|---|---|
| P27/ANI7 | PreDriver voltage measurement (input) |
| P10/TRDIOD1 | Port output or PWM output ($W_n$) |
| P11/TRDIOC1 | Port output or PWM output ($V_n$) |
| P12/TRDIOB1 | Port output or PWM output ($W_p$) |
| P13/TRDIOA1 | Port output or PWM output ($V_p$) |
| P14/TRDIOD0 | Port output or PWM output ($U_n$) |
| P15/TRDIOB0 | Port output or PWM output ($U_p$) |
| P55/INTP4 | ALARM signal input |
| P54 | Select for measurement of W phase Back EMF (electromotive force) |
| P53 | Select for measurement of V phase Back EMF |
| P52 | Select for measurement of U phase Back EMF |
| P51/SO00 | SPI data output for PreDriver control |
| P50/SI00 | SPI data input for PreDriver control |
| P30/SCK00 | SPI clock output for PreDriver control |
| P05 (CS) | SPI chip selection for PreDriver control |
| P06/TRJIO0 | System clock output for PreDriver |
| VSS | Ground voltage |
| VDD | Positive power supply |
| P146, P16/TRDIOC0, P17/TRDIOA0 | Unused terminal |
| P01 | LED output1 ON/OFF control |
| P00 | LED output2 ON/OFF control |
| P141 | LED output3 ON/OFF control |
| P140 | LED output4 ON/OFF control |
| P43 | LED output5 ON/OFF control |
| P42 | LED output6 ON/OFF control |
| P122 | For rotation direction command value input (CW/CCW) |
| P26/ANI6 | For rotation speed command value input (Analog value) |
| | Motor rotation start/stop command |
| P23/ANI3 | VM voltage measurement (input) |
| P03/RxD1 | UART input for ICS |
| P02/TxD1 | UART output for ICS |
| P31, P63, P62, P25/ANI5, P21/ANI1, P20/ANI0<br>P76 (CS), P75/SCK01, P74/SI01, P73/SO01<br>P60/SCLA0, P61/SDLA0 | Unused terminal |

List of interfaces of PreDriver of this system is shown in Table 2-3.

**Table 2-3 Port Interface (PreDriver)**

| Terminal name | Function |
|---|---|
| ISENADIN | PreDriver voltage output |
| TOINF | Motor control signal input ($W_n$) |
| TOINE | Motor control signal input ($V_n$) |
| TOINC | Motor control signal input ($W_p$) |
| TOINB | Motor control signal input ($V_p$) |
| TOIND | Motor control signal input ($U_n$) |
| TOINA | Motor control signal input ($U_p$) |
| ALARM | ALARM signal output |
| HIC_W | Select for measurement of W phase Back EMF |
| HIC_V | Select for measurement of V phase Back EMF |
| HIC_U | Select for measurement of U phase Back EMF |
| SI | Data input for SPI control |
| SO | Data output for SPI control |
| SCLK | Clock input for SPI control |
| CS | Chip select input for SPI control |
| CLK | System clock input |
| LD, TOINH, TOING | Unused terminal |
| VM | Power Supply |
| UHOUT | U phase High-Side Driver (Nch) driving output |
| U | For U phase detection |
| ULOUT | U phase Low-Side Driver (Nch) driving output |
| VHOUT | V phase High-Side Driver (Nch) driving output |
| V | For V phase detection |
| VLOUT | V phase Low-Side Driver (Nch) driving output |
| WHOUT | W phase High-Side Driver (Nch) driving output |
| W | For W phase detection |
| WLOUT | W phase Low-Side Driver (Nch) driving output |
| ISENP | Shunt resistance Plus side connection |
| ISENN | Shunt resistance Minus side connection |
| DrvGND | GND for the output stage circuit of PreDriver |
| VREG5 | Regulator Output (5V) |
| AGND | GND for Analog circuit of PreDriver |
| UH, VH, WH | Not use terminal |

### 2.2.2 Peripheral functions

List of peripheral functions used in this system is shown in Table 2-4.

**Table 2-4 Peripheral Functions List**

| Peripheral function | Usage |
|---|---|
| A/D converter | Rotation speed command value input (analog value) |
| | Voltage measurement (Back EMF measurement/VM voltage measurement) |
| | Current measurement |
| | Option: temperature measurement |
| General-purpose port | For rotation direction command value input (CW/CCW) |
| | Select of phase for Measurement of Back EMF |
| | Motor control signal output: port output |
| | LED output ON/OFF control |
| | Option: LED right output, toggle switch input / push switch input |
| Timer Array Unit | 500[us] interval timer |
| | Free run timer for rotation speed measurement |
| Timer RJ | System clock output for PreDriver |
| Timer RD | Motor control signal output: PWM output using complimentary PWM mode (six outputs) |
| External interruption | ALARM signal detection |
| Communication interface | SPI00 (for PreDriver control) |
| | UART1 (for ICS) |
| | option: SPI01, IICA0 |

(1) A/D converter

The rotation speed command value input (Analog value) and voltage are measured by using ‘A/D converter’.

A/D conversion is set channel selection mode to ‘Select mode’ and conversion operation mode to ‘One shot conversion mode’ (use software trigger).

Conversion speed of the A/D converter is 2.375[us] per channel and the smallest unit of conversion input value is shown in Table 2-5.

**Table 2-5 A/D converter**

| Item | Control value for A/D converter 1 bit | Channel |
|---|---|---|
| Rotation speed command input (analog value) | 5.56[rpm] step<br>(rotation speed range is 1100[rpm] to 4290[rpm] for both CW/CCW) | ANI6 |
| Voltage measurement | VM voltage measurement: 45.9[V] / 1024 = 0.045[V] | ANI3 |
| | Back EMF Note 3 measurement: 5[V] / 1024 = 0.0049[V] | ANI7 |
| Current measurement | Current Note 3 measurement: 200[A] / 1024 = 0.195[A] | ANI7 |

Note:

3. The Back EMF and current can be measured by switching a signal to be converting A/D by setting of ADC Selector Register (ADC_SEL) of the pre-driver side. The Back EMF can measure by sets "1" (Enable the measure of Back EMF) in BEMF_MODE_SEL of Hall Signal Processing Setting Register (HALL_SIG), and be sets 0x03 (Detection of BEMF Amp level) in ADC_SEL. The electric current measurement can reflect the control value by sets 0x01 (Detection of the current (ISENSE)) in ADC_SEL.

   Please refer to "RAJ306000 Series User’s Manual: Hardware (R18UZ0066EJ0100)" about the details.

(2) General-purpose port

Possible to select of Phase for Measurement of Back EMF to detect a Back EMF in a general-purpose port. When Detect Back EMF was operated, A potential difference of the chosen phase for Measurement of Back EMF and the Imaginary center tap (Common of motor) voltage converts by ADC, and a pre-driver judges whether higher than Imaginary center tap. Data of Back EMF can acquire from an A/D conversion level of ANI7 by set a general-purpose port of Measurement phase of Back EMF for detecting Back EMF to High.

In addition, Select of Phase for Measurement of Back EMF becomes effective by "1" (Select of Sensor-less) be set in a bit of HALL_MODE_SEL of Hall Signal Processing Setting Register (HALL_SIG) of the pre-driver.

Combination of Select of Phase for Measurement of Back EMF and general-purpose ports in this system is shown in Table 2-6.

**Table 2-6 Select of Phase for Measurement of Back EMF and General-purpose port**

| Terminal name | Select of phase for Measurement of Back EMF |
|---|---|
| P52 | U phase |
| P53 | V phase |
| P54 | W phase |

Also, this system output a motor control signal along with the PWM output using the port output function. Combination of Motor control signal output and general-purpose ports are shown in Table 2-7.

**Table 2-7 General-purpose port and motor control signal output**

| Terminal name | Motor control signal |
|---|---|
| P10/TRDIOD1 | $W_n$ |
| P11/TRDIOC1 | $V_n$ |
| P12/TRDIOB1 | $W_p$ |
| P13/TRDIOA1 | $V_p$ |
| P14/TRDIOD0 | $U_n$ |
| P15/TRDIOB0 | $U_p$ |

Note:

Please refer to "RL78/G1F User's Manual: Hardware (R01UH0516EJ0110)" about the notes when switching a general-purpose port from input mode to output mode,

(3) Timer Array Unit

・500[us] Interval timer

500[us] interval timer uses 'Interval timer function' of Timer Array Unit. In this system, channel 0 is used.

・Free-run timer for rotation speed measurement

Free-run timer for rotation speed measurement uses 'Interval timer function' of Timer Array Unit. However, it does not use the interruption. In this system, channel 1 is used.

Also, in this system, channel 2 and channel 3 are not used.

(4) Timer RJ

Using the pulse output mode, it outputs a 4 MHz square wave and supplies it as System clock for PreDriver.

(5) Timer RD

Using the Complementary PWM mode, it output (6-wire) a three-phase PWM with a triangle wave modulation and a short circuit preventive time.

In this system, support the PWM output of High active. (PWM frequency is 100[us]) In case of detect the ALARM (At the time of Input of Low signal to INTP4 port), PreDriver output signal will be change to Hi-Z (Output terminal value for Motor control signal becomes set to Low)

The combination of timer output and motor control signal are shown in Table 2-8.

**Table 2-8 timer output terminal and motor control signal output**

| Terminal name | Motor control signal |
|---|---|
| P10/TRDIOD1 | $W_n$ |
| P11/TRDIOC1 | $V_n$ |
| P12/TRDIOB1 | $W_p$ |
| P13/TRDIOA1 | $V_p$ |
| P14/TRDIOD0 | $U_n$ |
| P15/TRDIOB0 | $U_p$ |

(6) Interruption

List of interruptions in this system is shown in Table 2-9.

**Table 2-9 Interruption**

| Interruption name | Interruption source |
|---|---|
| P55/INTP4 | ALARM signal detection |
| INTTM00 | 500[us] Interval timer |
| INTTRD0 | Carrier frequency (PWM) |
| INTTRD1 | Carrier frequency (Underflow) |
| INTCSI00 | Complete of SPI00 communication for PreDriver control |

## 2.3 Software structure

### 2.3.1 Software file structure

Folders and files structure of the sample program is shown in Table 2-10 and Table 2-11.

**Table 2-10 Folder and Files Structure of Sample Program (1)**

| **RAJ306000_LESS_120_OPEN_CSP_CA_V103**<br>**RAJ306000_LESS_120_OPEN_CSP_CC_V103**<br>**RAJ306000_LESS_120_OPEN_E2S_CC_V103** | | |
|---|---|---|
| Inc | control_parameter.h | Header for control characteristic dependent processing part |
| | motor_parameter.h | Header for motor characteristic dependent processing part |
| | mtr_common.h | Header for Common definition |
| | mtr_ctrl_rl78g1f.h | Header for RL78/G1F dependent processing part |
| | mtr_ctrl_rl78g1f_t2001.h | Header for RL78/G1F & Board dependent processing part |
| | mtr_ctrl_t2001.h | Header for Board dependent processing part |
| | mtr_main.h | Main function, Header for user interface control |
| | mtr_spm_less_120_cpm.h | Header of Sensor-less 120-degrees conducting control dependent part |
| | r_dsp.h | Header for operation library |
| | r_less_120_is.h | Header for inductive sense library |
| | r_less_120_isw.h | Header for RL78/G1F dependent processing part |
| | r_stdint.h | Header for operation library |
| | version.h | Header of software revision |
| ics | ICS2_CA_RL78G1F.lib | Library for ICS (for CA78K0R) (Note 4) |
| | ICS2_CC_RL78G1F.lib | Library for ICS (for CC-RL) (Note 5) |
| | ics2_RL78G1F.h | Header for ICS |
| | RL78_vector.c | Interrupt handler for ICS |
| | RL78_vector.h | Interrupt handler header for ICS |
| lib | ICS2_CA_RL78G1F.lib | Library for ICS (for CA78K0R) (Note 4) |
| | ICS2_CC_RL78G1F.lib | Library for ICS (for CC-RL) (Note 5) |
| | r_less_120_is_ca.lib | Library for inductive sense (for CA78K0R) (Note 4) |
| | r_less_120_is_cc.lib | Library for inductive sense (for CC-RL) (Note 5) |
| src | mtr_ctrl_rl78g1f.c | RL78/G1F dependent processing part |
| | mtr_ctrl_rl78g1f_t2001.c | RL78/G1F & Board dependent processing part |
| | mtr_ctrl_t2001.c | Board dependent processing part |
| | mtr_interrupt.c | Interrupt handler |
| | mtr_main.c | Main function, user interface control |
| | mtr_spm_less_120_cpm.c | 120-degrees conducting control (using Sensor-less) dependent part |
| | r_less_120_isw.c | RL78/G1F dependent processing part |

Note:

4. "For CA78K0R" is included only in RAJ306000_LESS_120_OPEN_CSP_CA_V103.
5. "For CC-RL" is included only in RAJ306000_LESS_120_OPEN_CSP_CC_V103 and RAJ306000_LESS_120_OPEN_E2S_CC_V103.

**Table 2-11 Folder and Files Structure of Sample Program (2)**

| **RAJ306000_LESS_120_OPEN_CSP_CA_V103**<br>**RAJ306000_LESS_120_OPEN_CSP_CC_V103**<br>**RAJ306000_LESS_120_OPEN_E2S_CC_V103** | | |
|---|---|---|
| cg_src | r_cg_adc.c | RL78/G1F ADC processing |
| | r_cg_adc.h | RL78/G1F header of ADC processing |
| | r_cg_adc_user.c | RL78/G1F ADC processing (for User) |
| | r_cg_cgc.c | RL78/G1F clock output processing |
| | r_cg_cgc.h | Header for RL78/G1F clock output processing |
| | r_cg_cgc_user.c | RL78/G1F clock output processing (for User) |
| | r_cg_intp.c | RL78/G1F interrupt function processing |
| | r_cg_intp.h | Header for RL78/G1F interrupt function processing |
| | r_cg_intp_user.c | RL78/G1F interrupt function processing (for User) |
| | r_cg_macrodriver.h | Header for RL78/G1F Error definition |
| | r_cg_main.c | RL78/G1F main processing |
| | r_cg_main.h | Header for RL78/G1F main processing |
| | r_cg_port.c | RL78/G1F port function processing |
| | r_cg_port.h | Header for RL78/G1F port function processing |
| | r_cg_port_user.c | RL78/G1F port function processing (for User) |
| | r_cg_predrv.c | PreDriver processing |
| | r_cg_predrv.h | Header for PreDriver processing |
| | r_cg_predrv_prm.h | Header for PreDriver register parameter definition |
| | r_cg_predrv_reg.h | Header for PreDriver register address definition |
| | r_cg_predrv_user.c | PreDriver processing (for User) |
| | r_cg_sau.c | RL78/G1F Serial array unit processing |
| | r_cg_sau.h | Header for RL78/G1F Serial array unit processing |
| | r_cg_sau_user.c | RL78/G1F serial array unit processing (for User) |
| | r_cg_systeminit.c | RL78/G1F initial processing |
| | r_cg_tau.c | RL78/G1F timer array unit processing |
| | r_cg_tau.h | Header for RL78/G1F timer array unit processing |
| | r_cg_tau_user.c | RL78/G1F timer array unit processing (for User) |
| | r_cg_tmrd.c | RL78/G1F timer RD processing |
| | r_cg_tmrd.h | Header for RL78/G1F timer RD processing |
| | r_cg_tmrd_user.c | RL78/G1F timer RD processing (for User) |
| | r_cg_tmrj.c | RL78/G1F timer RJ processing |
| | r_cg_tmrj.h | Header for RL78/G1F timer RJ processing |
| | r_cg_tmrj_user.c | RL78/G1F timer RJ processing (for User) |
| | r_cg_userdefine.h | Header for RL78/G1F user definition |
| | r_cg_wdt.c | RL78/G1F watch dog timer processing |
| | r_cg_wdt.h | Header for RL78/G1F watch dog timer processing |
| | r_cg_wdt_user.c | RL78/G1F watch dog timer processing (for User) |

### 2.3.2 Module structure

Module structure of the sample program is described on Figure 2-4.

The relationship between module and file are shown in Table 2-12.

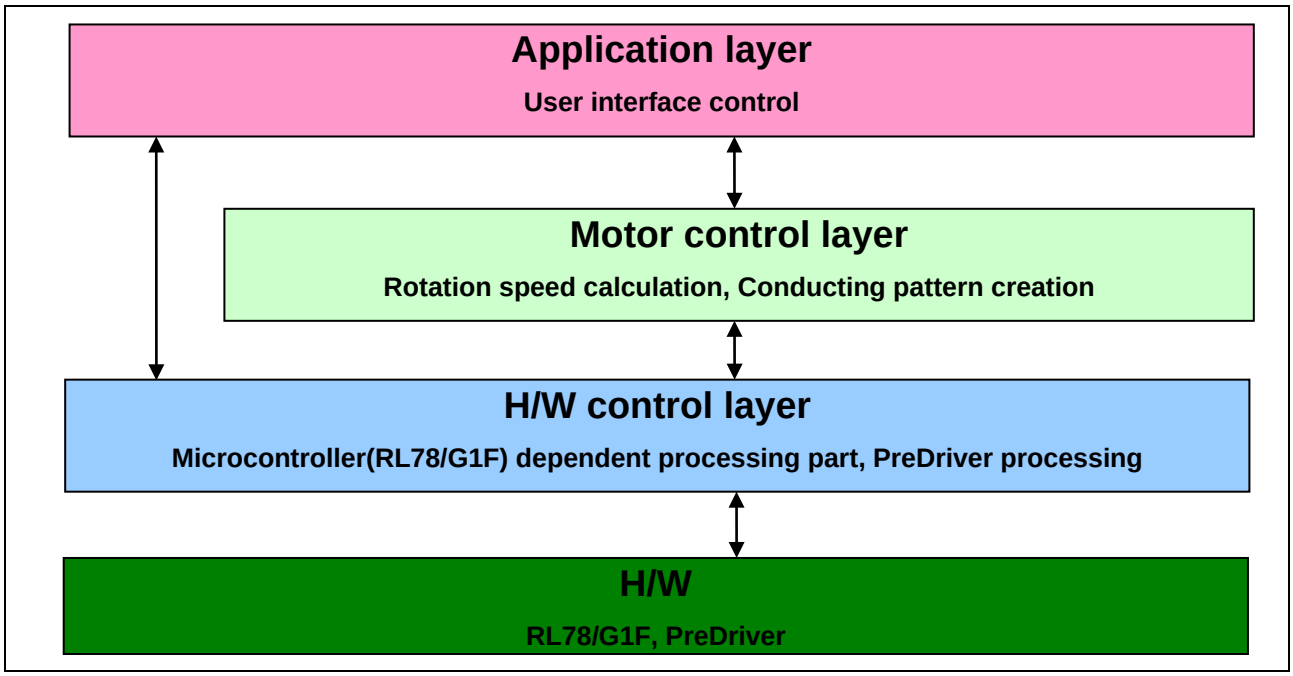


**Figure 2-4 Hierarchical Structure of Sample Program**

**Table 2-12 Hierarchical structure of Sample Program**

| | |
|---|---|
| **Application layer** | mtr_main.c |
| **Motor control layer** | mtr_interrupt.c, mtr_spm_less_120_cpm.c, r_less_120_is_*.lib |
| **H/W control layer** | mtr_ctrl_rl78g1f.c, mtr_ctrl_rl78g1f_t2001.c, mtr_ctrl_t2001.c, r_less_120_isw.c,<br>r_cg_adc.c, r_cg_adc_user.c, r_cg_cgc.c, r_cg_cgc_user.c,<br>r_cg_intp.c, r_cg_intp_user.c, r_cg_main.c,<br>r_cg_port.c, r_cg_port_user.c, r_cg_predrv.c, r_cg_predrv_user.c,<br>r_cg_sau.c, r_cg_sau_user.c, r_cg_systeminit.c,<br>r_cg_tau.c, r_cg_tau_user.c, r_cg_tmrd.c, r_cg_tmrd_user.c,<br>r_cg_tmrj.c, r_cg_tmrj_user.c, r_cg_wdt.c, r_cg_wdt_user.c |

## 2.4 Software specifications

Basic specifications of software of this system are shown in Table 2-13 and Figure 2-5.

**Table 2-13 Software Basic Specifications**

| Item | Content |
|---|---|
| Control method | 120-degrees conducting method |
| Motor rotation start/stop | Motor start/stop control is determined depending on the level of VR1(AIN6 terminal).<br>Input from ICS (Note 6) |
| Rotation direction control | Rotation direction command value (CW/CCW) control is determined depending on the level of SW1 (P122 terminal).<br>Input from ICS (Note 6) |
| Rotation Speed control | Rotation speed command value is determined from input voltage of VR1 (ANI6 terminal).<br>Control the PWM duty proportional to the rotation speed command value with 0 to 100[%] |
| Rotation speed control range | 1100[rpm] to 4290[rpm] for both CW/CCW |
| Detecting of magnetic pole position of rotor of electric motor | Every electric angle position of 30 degrees from Zero Cross position of Back EMF. (Every electric angle 60 degrees) |
| Carrier frequency (PWM) | 10[KHz] |
| Control cycle/Rotation speed operation | Every electric angle position of 30 degrees from Zero Cross position of Back EMF. (Every electric angle 60 degrees)<br>・Determination of PWM duty setting and Conducting pattern<br>・Calculate rotation speed from the difference value with the counter level of before 1 rotation (360 degrees) |
| Processing stop for protection | Output terminal of Motor control signal is set to Low state at the time of detect the below errors.<br>・ALARM error<br>・Over voltage error<br>・Rotation speed abnormal error<br>・Timeout error<br>・Current pattern error<br>・Error of detection for Back EMF<br>・Under voltage error |

Note:

6. Please refer to the "4 Development support tool In Circuit Scope" about details.

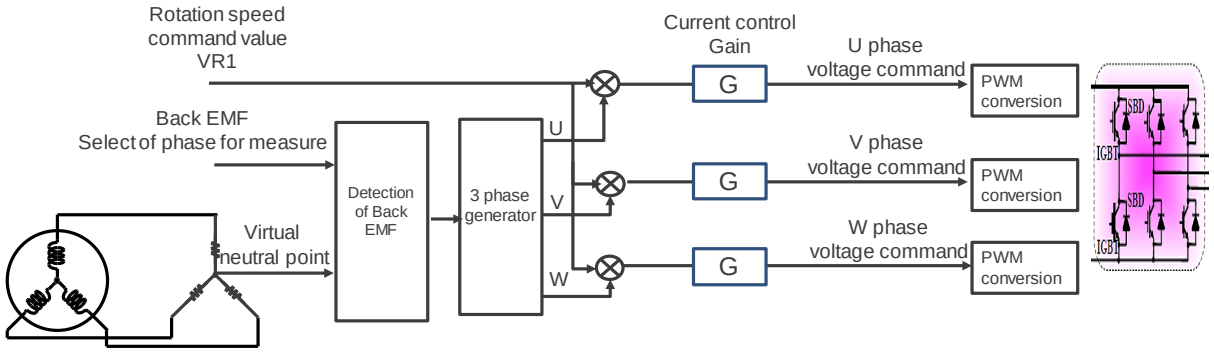


**Figure 2-5 Basic specification of Software**

# 3. Descriptions of control program

The target sample programs of this application note are explained here.

## 3.1 Contents of control

### 3.1.1 Motor start / stop

Starting and stopping of the motor are controlled by input from VR1 and SW1. An analog input port (ANI6) is assigned to VR1. The input is A/D converted within the main loop to calculate Rotation speed command value. Program is judged that Motor was started at the time of the command value is more than 1200[rpm]. and Program is judged the motor was stopped at the time of the command value is less than 1100[rpm].

General-purpose port (P122 terminal) is assigned to SW1 and, in main loop, acquires a High/Low state of the P122 terminal and assumes it a rotation direction command value. The rotation direction is judged from a rotation direction command value.

### 3.1.2 Rotation direction command value, Rotation speed command value, VM voltage.

(1) Rotation direction command value

Rotation direction command value can be set by high/low state of SW1 or input information from ICS.

(2) Rotation speed command value

Rotation speed command value can be set by A/D conversion of the VR1 output value (Analog value). Set VR1 output value converted A/D to the rotation speed command value.
VR1 value that A/D converted is used to Rotation speed command value as shown below (Table 3-1).

**Table 3-1 Conversion Ratio of the Rotation Speed Command Value**

| Item | Conversion ratio (Command value: A/D conversion value) | Channel |
|---|---|---|
| Rotation speed command value | 1100[rpm] to 4290[rpm]: 03FFH to 0000H | ANI6 |

(3) VM voltage

It is used for detection of over voltage and under voltage. (When an abnormality is detected, PWM is stopped.) Conversion ratio of VM voltage value is shown in Table 3-2.

**Table 3-2 Conversion Ratio of VM Voltage**

| Item | Conversion ratio (VM voltage: A/D conversion value) | Channel |
|---|---|---|
| VM voltage | 0.0[V] to 45.9[V]: 0000H to 03FFH | ANI3 |

### 3.1.3 Rotation speed calculation

After having let channel 1 of the timer array unit make a free run, a counter level of the timer at the time of a position of 30 degrees electric angle from Zero Cross position of Back EMF is acquired, and calculate a rotation speed of motor from the difference share of the counter level before 1 rotation (360 degrees). Additional, Processing of LPF (low-pass filter) is carried out about this calculation result data. Calculation Method (Conceptual diagram) of rotation speed is shown in Figure 3-1.

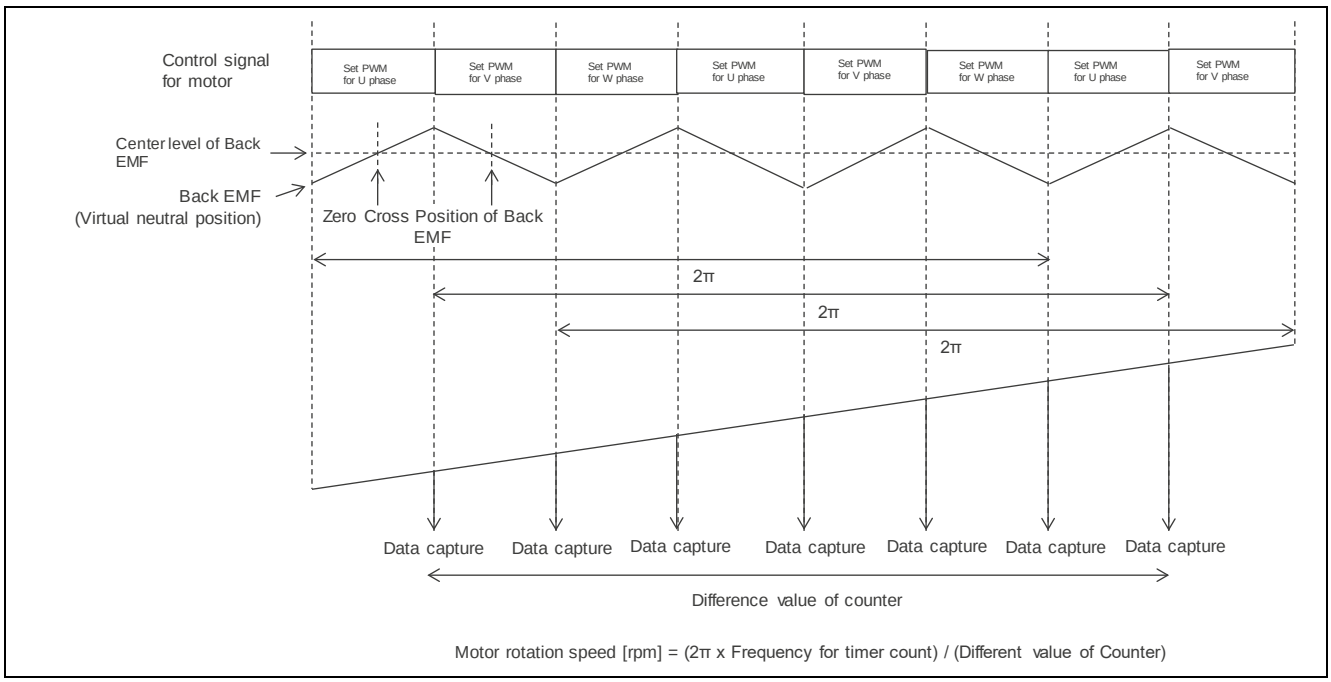


**Figure 3-1 Rotation Speed Calculation Method**

### 3.1.4 Voltage control by PWM

PWM control is used for the output voltage control. The PWM control is a control method that continually adjusts the average voltage by varying the duty of pulse, and PWM control is controlled by value that PWM duty value is proportional to Rotation speed command value. Conception diagram of the PWM control is shown Figure 3-2.

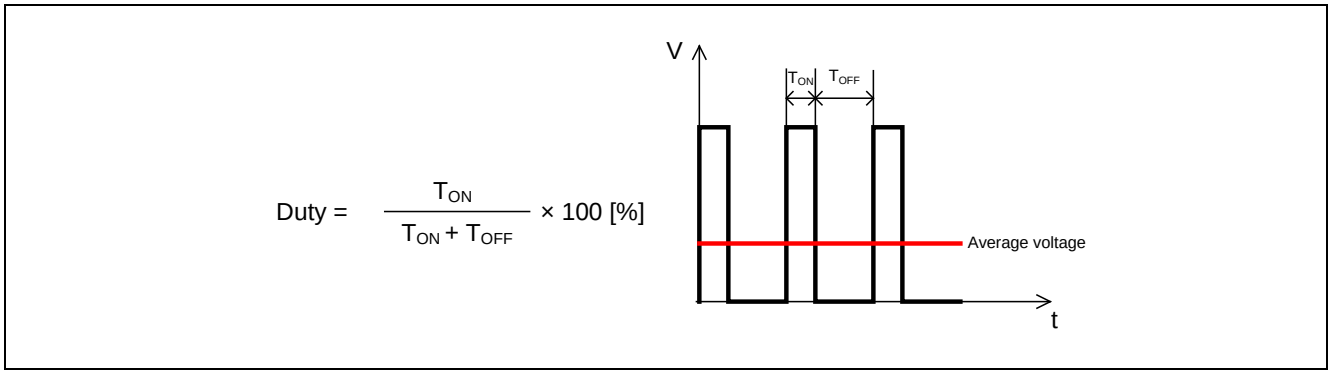


**Figure 3-2 PWM control**

Chopping control is adopted at the first 60 degrees in this system and output voltage and speed are controlled. An example of motor control signal output waveforms at the time of complementary first 60 degrees chopping is shown in Figure 3-3.

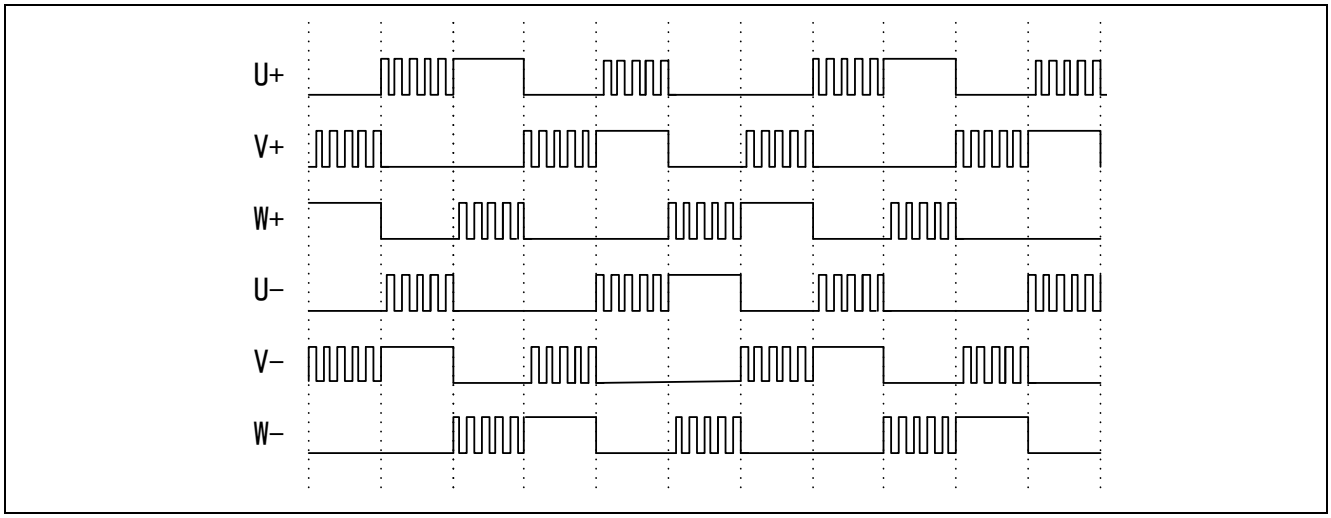


**Figure 3-3 Complimentary first 60 degrees chopping**

### 3.1.5 Back EMF detection and Current detection

The Back EMF detection method supports following two ways. As for one, when PWM Duty level is less than 50[%], A/D conversion is executed the Back EMF detection (Regeneration period) in a time regeneration of Motor. Another, when PWM Duty level is more than 50[%], A/D conversion is executed the Back EMF detection (Current period) in a time drive of Motor. Back EMF detection timing is shown in Figure 3-4 and Figure 3-5.

Because the current detection is outputted data with a terminal same as Back EMF detection. It is necessary to switch a circuit function of the pre-driver in ADC_SEL (ADC Selector Register). After this, switch to Back EMF Detection in ADC_SEL and prepare for Back EMF detection. Since, switch ADC_SEL of Current detection and Back EMF detection and measure at the time of rotate the motor. This change has to carry out without depending on the level of PWM Duty. Current detection timing is shown in Figure 3-4 and Figure 3-5.

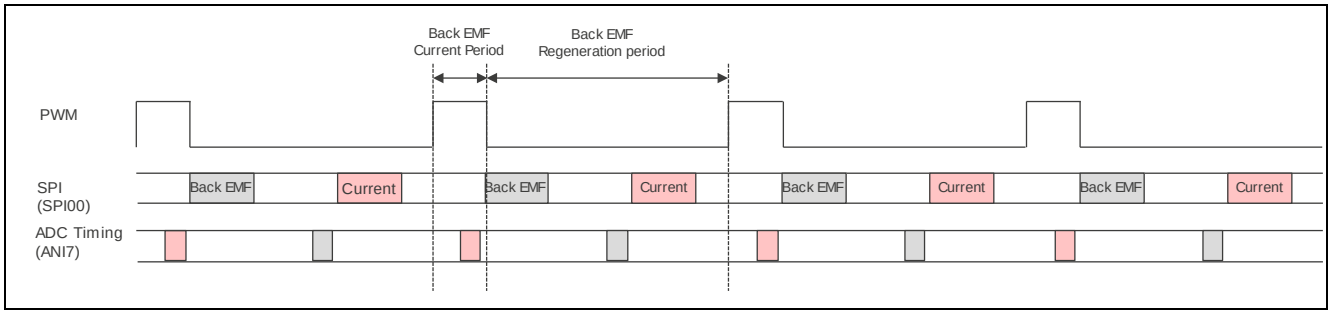


**Figure 3-4 Back EMF detection (Regeneration period) and timing of Current detection (PWM Duty < 50[%])**

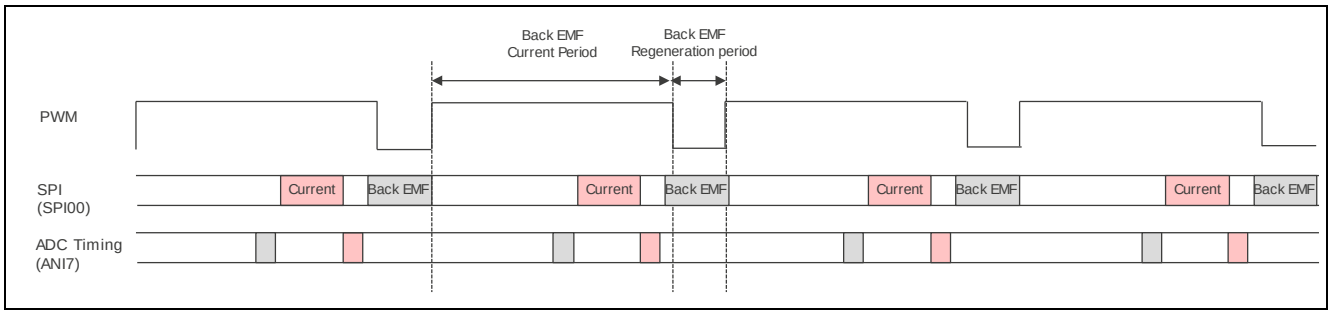


**Figure 3-5 Back EMF detection (Current period) and timing of Current detection (PWM Duty >= 50[%])**

### 3.1.6 State transition

State transition diagrams of the sample programs are shown in Figure 3-6.

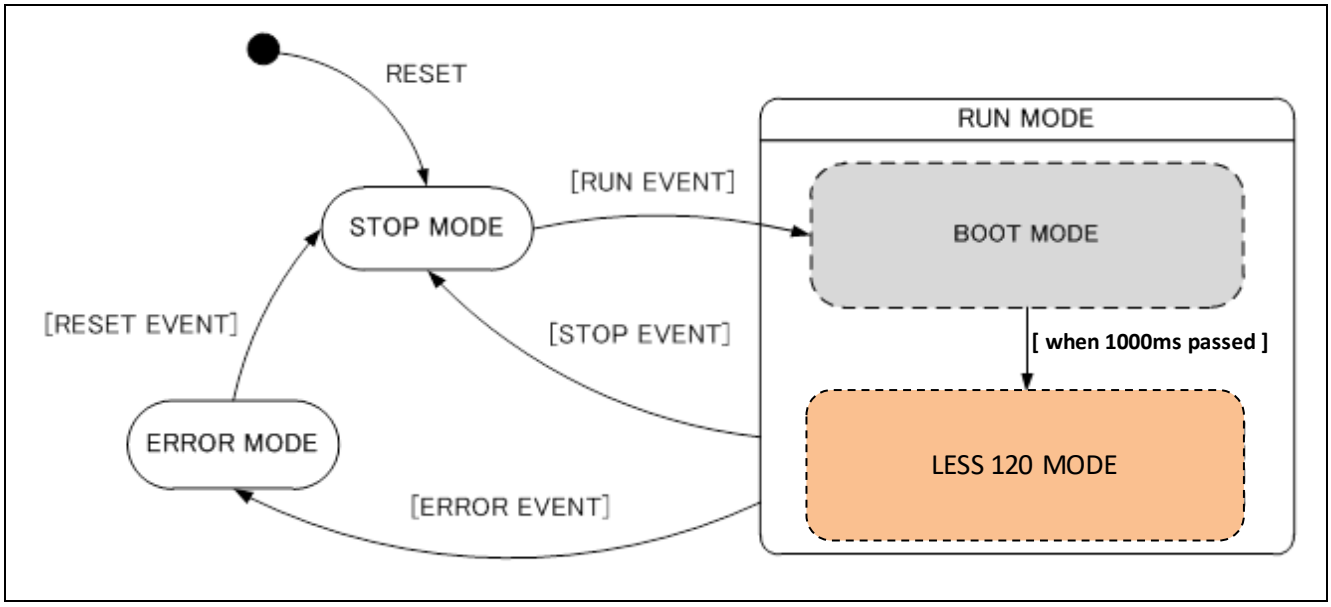


**Figure 3-6 State Transition Diagram**

### 3.1.7 Start method of Motor by Sensor-less

Sensor-less 120-degrees control is based on the estimation of the position of the magnetic pole at every 60 degrees using Back EMF by the change of the magnetic flux of the permanent magnet (rotor). However, the Back EMF is generated by rotation of the motor. Therefore, it is necessary to detect the position of the magnetic pole without using Back EMF during the motor startup.

Therefore, Inductive Sense sequence is used by which magnetic pole position detects by injecting an electricity pattern. Same concept is used during motor stops where the position of the permanent magnet is detected independent of Back EMF.

Once the Back EMF is induced, the control switches from inductive sense sequence to Back EMF based control, Back EMF of each phase is detected, and the phase of the Hi-Z state is judged. The detection control of the Hi-Z state is carried out, and an electricity pattern is decided. Back EMF detection (motor drive) is carried out by this electricity pattern information.

Zero cross position of Back EMF compares the center level of Back EMF with Back EMF, and it is determined. In a timing to change it to the electricity pattern of next a phase is decided by calculating the time of the electric angle 30 degrees position from the Zero cross position of the Back EMF. Because this sequence continuous operation after this time passed, the rotation control of the motor is carried out.

U phase is detected in the detection movement of the Hi-Z state by a sequence of Back EMF, and the example which a motor rotation is controlled is shown in Figure 3-7 start methods.

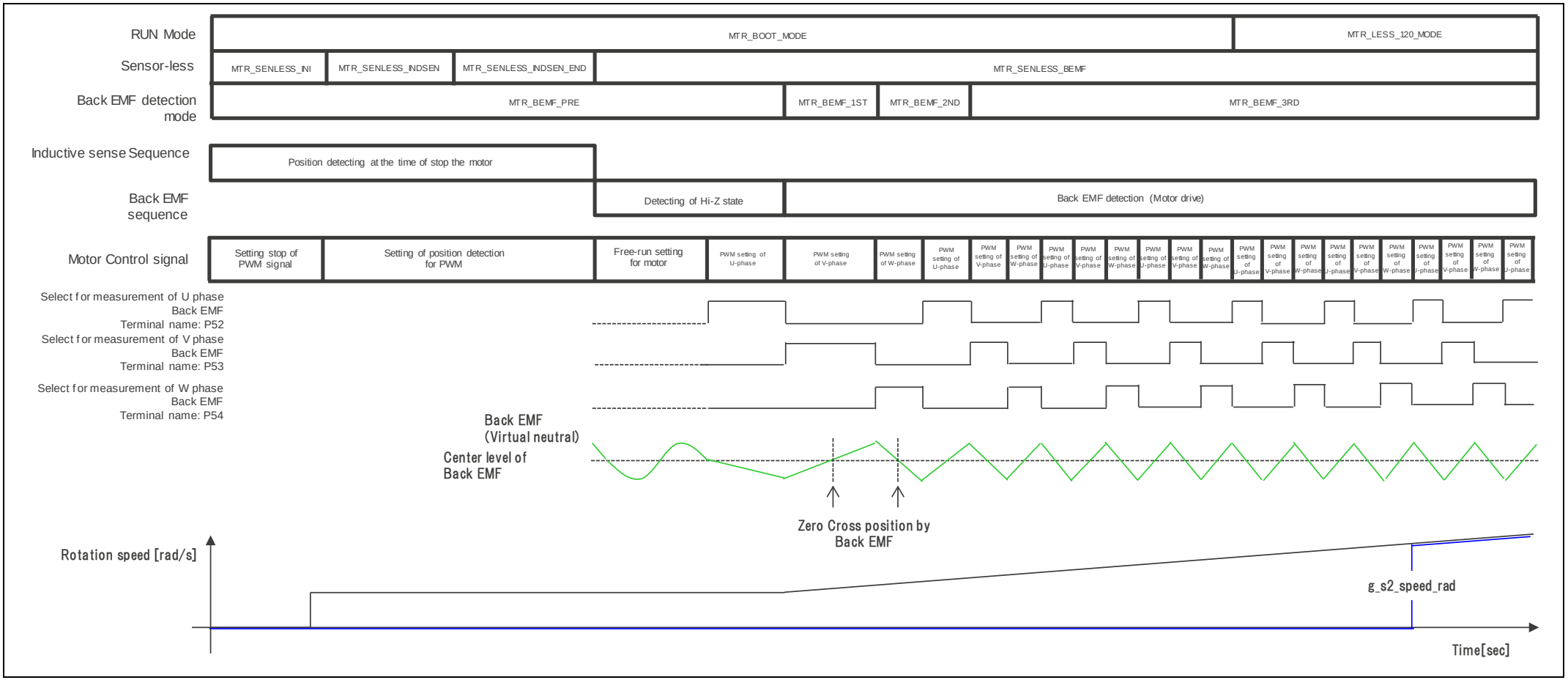


**Figure 3-7 Example for Motor rotation control**

### 3.1.8 System protection function

This system has below error condition. Emergency stop function is operate as each condition of the following. Each set value related to the system protection function is shown in Table 3-3.

・ALARM error

Emergency stop is performed by setting the output of PreDriver to the high impedance state (Output terminal signal for motor control is Low state) by the emergency stop signal (ALARM detection) from PreDriver.

・Over voltage error

When an over voltage is detected (when the voltage exceeds the limit value) in VM voltage on a cycle of over voltage detection, System is performed an emergency stop.

・Rotation speed abnormality error

When the rotation speed exceeded the limit value on a cycle of the rotation speed detect operation, System is performed an emergency stop.

・Timeout error

When an interval time by a switch of Motor control signal is not switch for over Timeout limit value at Time out error monitoring period, System is performed an emergency stop.

・Current pattern error

When detecting an error pattern while monitoring the current pattern every time the motor control signal is switched, System is performed an emergency stop.

・Back EMF detection error

Even if the processing time of Back EMF sequence exceeds time-out time in a monitoring period of Back EMF detection error, when sequence is not shifted, a system urgently stops.

・Under voltage error

When an under voltage is detected (when the voltage less than the limit value) in VM voltage on a cycle of under voltage detection, System is performed an emergency stop.

**Table 3-3 Setting value for Protect function of each system**

| Error Condition | | Setting value |
|---|---|---|
| Over voltage error | Over voltage limit value | 30[V] |
| | Monitoring interval | 100[us] |
| Rotation speed abnormality error | Rotation speed limit value | 4290[rpm] |
| | Monitoring interval | 100[us] |
| Timeout error | Timeout setting | 20[ms] |
| Back EMF detection Error | Timeout setting | 250[ms] |
| | Monitoring interval | 100[us] |
| Under voltage error | Under voltage limit value | 6[V] |
| | Monitoring interval | 50[us] |

### 3.1.9 System protect function (PreDriver safety function)

The PreDriver safety function can be enabled / disabled with the ALARM operation setting register (ALMOPE).

Please refer to the data sheet about details.

## 3.2 Function specifications

Lists of control functions are shown in Table 3-4 and Table 3-5.

**Table 3-4 List of Control Functions (1)**

| File name | Function overview | Processing overview |
|---|---|---|
| mtr_main.c | main()<br>input: none<br>output: none | ・Hardware initialization function call<br>・User interface initialization function call<br>・Main function use variable initialization function call<br>・Status transition and event execution function call<br>・Main function<br>- Main processing execution function call<br>- Execution of Sensor-less processing<br>- Watchdog timer clear function call |
| | ctrl_ui()<br>input: none<br>output: none | ・Change Motor status<br>・Determination of rotation speed command value and rotation direction command value |
| | ics_ui()<br>input: none<br>output: none | ・Change Motor status<br>・Determination of rotation speed command value and rotation direction command value |
| | ctrl_led()<br>input: none<br>output: none | Control the output pattern of ON/OFF for LED |
| | ics_predrv_reg_ctrl()<br>input: none<br>output: none | Control for PreDriver register read/write from ICS |
| | mcu_sw_init()<br>input: none<br>output: none | Initialization of F/W<br>・initialization of inductive sense library<br>・initialization of F/W variables<br>・initialization of ICS<br>・initialization of sequence processing<br>・execution of RESET event |
| | software_init()<br>input: none<br>output: none | Initialization of variables used in the main function |
| | ctrl_sensorless()<br>input: none<br>output: none | Sensor-less processing<br>・Main process of inductive sense<br>・Start-up of sequence for Back EMF<br>・Setting of Motor drive control |
| mtr_ctrl_rl78g1f.c | clear_wdt()<br>input: none<br>output: none | Clear Flag for the watchdog timer |
| | mtr_clear_oc_flag()<br>input: none<br>output: none | Clear Flag for the pulse output forced shutdown |
| | mtr_clear_trd0_imfa()<br>input: none<br>output: none | Clear Flag for the TRD0 Compare match (IMFA) |
| | mtr_clear_trd1_udf()<br>input: none<br>output: none | Clear Flag for the TRD1 Underflow (UDF) |

| | | |
|---|---|---|
| mtr_ctrl_rl78g1f_t2001.c | mtr_ctrl_start()<br>input: none<br>output: none | Motor startup processing |
| | mtr_ctrl_stop()<br>input: none<br>output: none | Motor stop processing |
| | mtr_change_pattern()<br>input: Conduction pattern<br>output: none | Change the motor control signal output<br>・Setting a conducting pattern<br>・Setting select of phase for measurement of Back EMF<br>・Setting of the kind of the conducting pattern by PWM Duty<br>・Changing the motor status when a conducting<br>pattern error occurs<br>・Event processing selection function call |
| | mtr_get_adc()<br>input: A/D channel<br>output: A/D conversion result | Processing execution of the A/D convert |
| mtr_ctrl_t2001.c | get_vr1()<br>input: none<br>output: A/D conversion result of VR1 | Obtain of the A/D conversion value of the trigger level |
| | led_on()<br>input: LED channel number<br>output: none | Turning LED ON |
| | led_off()<br>input: LED channel number<br>output: none | Turning LED OFF |
| mtr_interrupt.c | mtr_alarm_interrupt()<br>input: none<br>output: none | ALARM interrupt processing<br>・Change motor status<br>・Function call for selection of an event processing<br>・Function call for clear of flag of a forced interception of the pulse output |
| | mtr_tau0_interrupt()<br>input: none<br>output: none | 500[us] interrupt processing<br>・Judgement processing for switching Operation mode. |
| | mtr_carrier_interrupt()<br>input: none<br>output: none | Interrupt processing by a carrier frequency<br>・A/D conversion of detection data for Back EMF value and Current value<br>・SPI communication for switching of detection for Back EMF value and Current value<br>・Detection sequence for Back EMF<br>・Setting for the conducting pattern<br>・Waiting for motor rotation stop<br>・Function call for Error check |
| | mtr_carrier_udf_interrupt()<br>input: none<br>output: none | Underflow interrupt processing by a carrier frequency<br>・A/D conversion of detection data for Back EMF value and Current value<br>・SPI communication for switching of detection for Back EMF value and Current value |

| | | |
|---|---|---|
| mtr_spm_less_120_cpm.c | R_MTR_InitSequence()<br>input: none<br>output: none | Initialization of sequence processing |
| | R_MTR_ExecEvent()<br>input: occurred event<br>output: none | ・Execute to change the status.<br>・ Call execution function of suitable processing for the event. |
| | mtr_act_run()<br>input: motor status<br>output: motor status | ・Variable initialization function call upon motor startup<br>・Motor control startup function call |
| | mtr_act_stop()<br>input: motor status<br>output: motor status | Motor control stop function call |
| | mtr_act_none()<br>input: motor status<br>output: motor status | No processing is performed. |
| | mtr_act_reset()<br>input: motor status<br>output: motor status | Initialization of Global variable for return from Error state. |
| | mtr_act_error()<br>input: motor status<br>output: motor status | Motor control stop function call at the time of Error occur. |
| | mtr_pattern_set()<br>input: Current pattern<br>output: none | Setting for conducting pattern<br>・Rotation speed measurement function call<br>・Determine of Conducting pattern<br>・Motor control signal output change function call |
| | mtr_convert_indsenpat_to_pwmpat()<br>input: Current pattern<br>output: Current pattern (Conversion data) | Processing for switching of Current pattern |
| | mtr_get_bemf_threshold()<br>input: none<br>output: Center value of Back EMF | Processing for acquire a center value of Back EMF |
| | mtr_get_bemf_threshold_direct()<br>input: Classification of Back EMF detection. Classification of Hi-Z<br>output: Center value of Back EMF | Processing for acquire a center value of Back EMF |
| | mtr_get_bemf_dir()<br>input: none<br>output: Direction of Back EMF detection | Processing for acquire a direction data of Back EMF detection |
| | mtr_set_predrv_isense()<br>input: none<br>output: none | Processing for setting of current detection |
| | mtr_set_predrv_bemf()<br>input: none<br>output: none | Processing for setting of Back EMF detection |
| | mtr_speed_calc()<br>input: none<br>output: none | Processing of calculation for rotation speed measurement |
| | mtr_start_init()<br>input: none<br>output: none | Initialization of the variable that required at the time of motor startup |
| | mtr_set_variables()<br>input: none<br>output: none | Set Input data at ICS to Protecting variable. |
| | R_MTR_IcsInput()<br>input: structure of ICS variables<br>output: none | Obtaining of data that inputted from the ICS. |

| | | |
|---|---|---|
| | R_MTR_SetSpeed()<br>input: rotation speed command value<br>output: none | Rotation speed setting |
| | R_MTR_SetDir()<br>input: rotation direction command value<br>output: none | Rotation direction setting |
| | R_MTR_GetSpeed()<br>input: none<br>output: rotation speed information | Obtaining the rotation speed |
| | R_MTR_GetDir()<br>input: none<br>output: rotation direction information | Obtaining the rotation direction. |
| | R_MTR_GetStatus()<br>input: none<br>output: motor status | Obtaining the motor status |
| | mtr_error_check()<br>input: none<br>output: none | Monitoring and Detection of Error |
| r_less_120_isw.c | r_isw_set_port()<br>input: general port setting<br>output: processing result | Set general purpose port ($U_p$, $U_n$, $V_p$, $V_n$, $W_p$, $W_n$) |
| | r_isw_wait()<br>input: specified time<br>output: processing result | Wait for a specified time [us] |
| | r_isw_clear_wdt()<br>input: none<br>output: processing result | Clear watchdog timer |
| | r_isw_get_adc()<br>input: none<br>output: processing result, A/D conversion result | Get A/D conversion result (current) |
| r_less_120_is_*.lib | R_IS_Init()<br>input: none<br>output: processing result | Initialization of library |
| | R_IS_GetRevision()<br>input: none<br>output: processing result, library revision | Get library revision |
| | R_IS_Main()<br>input: initial values<br>output: processing result, stop position | Main process of inductive sense<br>Execute inductive sense to detect the stop position |

**Table 3-5 List of Control functions (2)**

| File name | Function overview | Processing overview |
|---|---|---|
| r_cg_adc.c | R_ADC_Create()<br>input: none<br>output: none | Initialization of A/D converter |
| r_cg_adc_user.c | r_adc_interrupt()<br>input: none<br>output: none | SPI communication ADC mode SPI start judgement |
| r_cg_cgc.c | R_CGC_Create()<br>input: none<br>output: none | Initialization of clock frequency (CGC) |
| r_cg_intp.c | R_INTP_Create()<br>input: none<br>output: none | Initialization of external interrupt (INTP) |
| r_cg_main.c | R_MAIN_UserInit()<br>input: none<br>output: none | PreDriver startup processing |
| r_cg_port.c | R_PORT_Create()<br>input: none<br>output: none | Initialization of I/O port setting |
| r_cg_predrv.c | predriver_hw_init()<br>input: none<br>output: none | PreDriver initialization setting |
| | R_PREDRV_TRIM_Create()<br>input: none<br>output: SPI status | PreDriver trimming data setting |
| | R_PREDRV_InitSequence()<br>input: none<br>output: none | PreDriver initialization processing |
| | R_PREDRV_<br>ErrorRecoverySequence()<br>input: ALARM status<br>output: none | ALARM recovery processing |
| r_cg_predrv_user.c | R_PreDrvReg_Read()<br>input: read address<br>output: SPI status, read data | Read processing to PreDriver register |
| | R_PreDrvReg_Write()<br>input: write address, write data<br>output: SPI status | Write processing to PreDriver register |
| | R_PreDrvReg_Write_Intr_Mode()<br>input: write address, write data<br>output: SPI status | Write processing to PreDriver register<br>INTR Mode |
| | R_PreDrv_Set_AdcSel()<br>Input: Indicate to select of ADC_SEL<br>Output: none | Setting processing of ADC_SEL |
| | R_PreDrv_Set_MotEn()<br>Input: Indicate to Motor drive control<br>Output: none | Setting processing of Motor drive control |

| | | |
|---|---|---|
| r_cg_sau.c | R_SAU0_Create()<br>input: none<br>output: none | Initialization of serial array unit (SAU) |
| | R_UART1_Create()<br>input: none<br>output: none | Initialization of UART1 |
| | R_CSI00_Create()<br>input: none<br>output: none | Initialization of SPI communication (for PreDriver communication) |
| | R_CSI00_Start()<br>input: none<br>output: none | Startup SPI communication (for PreDriver communication) |
| | R_CSI00_<br>Send_Receive_SPI_mode()<br>input: tx buffer<br>buffer size<br>rx buffer<br>SPI mode<br>output: SPI status | SPI communication processing |
| r_cg_sau_user.c | r_csi00_interrupt()<br>input: none<br>output: none | SPI interrupt processing (for PreDriver communication) |
| r_cg_systeminit.c | hdwinit()<br>input: none<br>output: none | Initial setting of H/W |
| r_cg_tau.c | R_TAU0_Create()<br>input: none<br>output: none | Initialization of TAU |
| r_cg_tmrd.c | R_TMRD0_Create()<br>input: none<br>output: none | Initialization of Timer RD (TRD) |
| | R_TMRD0_Start()<br>input: none<br>output: none | PWM output start |
| r_cg_tmrj.c | R_TMRJ0_Create()<br>input: none<br>output: none | Initialization of Timer RJ (TRJ) |
| | R_TMRJ0_Start()<br>input: none<br>output: none | Start supply a clock for PreDriver |
| r_cg_wdt.c | R_WDT_Create()<br>input: none<br>output: none | Initialization of Watch dog timer |

## 3.3 Specification of variables

Lists of variables for the sample program are shown in Table 3-6.

**Table 3-6 List of Variables**

| Variable name | Type | Content | Remarks |
|---|---|---|---|
| g_s2_max_speed | int16_t | Rotation speed command maximum value | Mechanical angle [rpm] |
| g_s2_min_speed | int16_t | Rotation speed command minimum value | Mechanical angle [rpm] |
| g_s2_margin_min_speed | int16_t | Rotation speed command minimum value for motor stop | Mechanical angle [rpm] |
| g_s2_ref_speed | int16_t | Setting of rotation speed by user | Mechanical angle [rpm] |
| g_u2_speed_rpm | uint16_t | Rotation speed calculation value | Mechanical angle [rpm] |
| g_u1_rot_dir | uint8_t | Setting of rotation direction by user | 0: CW<br>1: CCW |
| g_u1_motor_status | uint8_t | Management of motor status by user | 0: Stop<br>1: Rotating<br>2: Error |
| g_u1_stop_req | uint8_t | Motor stop command flag | Stop is determined when the rotation speed command value is less than 1100[rpm] |
| g_u1_pdrv_status | uint8_t | Register Read/Write for PreDriver<br>Error status | - |
| g_u1_err_recovery_req | uint8_t | ALARM recovery processing request flag | 0: Disable<br>1: Enable |
| g_u1_get_alarm_sts1 | uint8_t | PreDriver register ALMSTS1 acquired value | - |
| g_u1_store_alarm_sts1 | uint8_t | PreDriver register ALMSTS1 stored value | - |
| g_u1_get_alarm_sts2 | uint8_t | PreDriver register ALMSTS2 acquired value | - |
| g_u1_store_alarm_sts2 | uint8_t | PreDriver register ALMSTS2 stored value | - |
| g_u2_fw_revision | uint16_t | F/W Revision information | F/W Version information (103) |
| g_u2_is_lib_revision | uint16_t | Revision information of Library for inductive sense | Version information of Library for inductive sense (104) |
| g_s2_sw_userif | int16_t | Flag for switch of Board UI | 0: not use Board UI<br>1: use Board UI |
| g_s2_mode_system | int16_t | Flag for system mode | 0: Stop 1: Motor startup<br>2: Error 3: Reset |
| g_s2_enable_write | int16_t | Flag for Write enable of ICS | Toggle operate |
| ics_input | MTR_ICS_INPUT | Structure for input of ICS | - |
| g_u2_cnt_boot_mode | uint16_t | Counter for time measurement of Boot mode | 500[us] after motor startup is counted. |
| g_u2_cnt_wait_stop | uint16_t | Motor rotation stop waiting counter | 10[ms] after motor stop processing is counted. |
| g_u1_flg_wait_stop | uint8_t | Flag for waiting time of Motor rotation stop | The flag is set upon receive the stop command of the motor.<br>After the stop processing of motors, this flag is cleared at the time of non-change of the motor control signal between 10[ms]. |
| g_u1_enable_write | uint8_t | Flag for Write enable of Structure for ICS input. | 0: Disable<br>1: Enable |
| g_s2_vdc_ad | int16_t | A/D value of VM voltage | [V] |
| g_s2_reci_vdc_ad | int16_t | Inverse of A/D value from VM voltage | - |
| g_s2_pdrv_ad | int16_t | A/D value of PreDriver voltage | [V] |

| | | | |
|---|---|---|---|
| g_s2_pwm_duty | int16_t | Setting value of Timer RD compare register | PWM Duty setting value: 0 to 3198<br>PWM Duty [%]: 0 to 100 |
| g_s2_ref_speed_rad | int16_t | Rotation speed command value | Electrical angle (Scale: Q2) [rad/s] |
| g_s2_speed_rad | int16_t | Rotation speed calculation value | Electrical angle (Scale: Q2) [rad/s] |
| g_s2_speed_lpf_k | int16_t | Rotation speed LPF parameter | - |
| g_u1_cnt_ics | uint8_t | Counter for interval of ICS function call | - |
| g_u2_run_mode | uint16_t | Operation mode management | 0: Boot mode<br>3: Normal operation (LESS 120) mode |
| g_u1_error_status | uint8_t | Error status management | 0x01: ALARM error<br>0x02: Over voltage error<br>0x04: Rotation speed abnormality error<br>0x08: Timeout error<br>0x10: Current pattern error<br>0x20: Back EMF detection error<br>0x40: Under voltage error<br>(0x80: Undefined error) |
| g_u1_mode_system | uint8_t | Management of system mode | 0: Stop, 1: Run, 2: Error |
| g_u1_v_pattern | uint8_t | Conducting pattern | - |
| g_u2_cnt_timeout | uint16_t | Stop determination time measurement counter | Cleared when the conducting pattern is switched. |
| g_u1_direction | uint8_t | Rotation direction management | 0: CW<br>1: CCW |
| g_u2_less_timer_cnt | uint16_t | Free run timer count value | TCR01 |
| g_u2_pre_less_timer_cnt | uint16_t | Previous of free run timer count value | - |
| g_s2_timer_cnt_ave | int16_t | Rotation speed measurement timer count difference for 2 Pi | - |
| g_u2_timer_cnt_buf | uint16_t | Buffer of a timer count for measurement of Rotation speed | - |
| g_u2_timer_cnt_num | uint16_t | Buffer numbers of a timer count for measurement of Rotation speed | - |
| ics_input_buff | MTR_ICS_INPUT | ICS input variable structure | - |
| g_s2_ref_speed_rpm_vr1 | int16_t | Rotation speed command value | Mechanical angle [rpm] |
| g_u1_alarm_sts1 | uint8_t | PreDriver register ALMSTS1 stored value | For ICS display |
| g_u1_alarm_sts2 | uint8_t | PreDriver register ALMSTS2 stored value | For ICS display |
| g_u1_PreDriver_error | uint8_t | PreDriver sequence error status | ・PreDriver initial sequence<br>・ALARM recovery sequence |
| g_spi00_comend_flag | uint8_t | SPI communication condition flag | TURE: communication end<br>FALSE: connecting |
| g_spi00_adcend_flag | uint8_t | SPI communication ADC End flag | TURE: ADC end<br>FALSE: ADC executing |
| g_spi00_commode | uint8_t | SPI communication mode | - |
| gp_csi00_rx_address | uint8_t | SPI communication receives data address | Obtain of PreDriver register value |
| g_csi00_rx_length | uint16_t | SPI communication receives data length | - |
| g_csi00_rx_count | uint16_t | SPI communication receives counter | - |
| gp_csi00_tx_address | uint8_t | SPI communication transmission data address | Designation of PreDriver register address |
| g_csi00_send_length | uint16_t | SPI communication transmission data length | - |
| g_csi00_tx_count | uint16_t | SPI communication transmission counter | - |
| g_u1_sensorless_mode | uint8_t | Management of Sensor-less mode | - |
| g_u1_bemf_mode | uint8_t | Management of Back EMF detection mode | - |
| g_u1_bemf_seq | uint8_t | Management of sequence number for Back EMF | - |
| g_u1_indsen_signal | uint8_t | Position of Inductive sensing pattern | Current pattern |
| g_s2_bemf_ad_pre | int16_t | A/D conversion value of Back EMF detection value on the last time | - |

| | | | |
|---|---|---|---|
| g_s2_bemf_ad_now | int16_t | A/D conversion value of Back EMF detection value on this time. | - |
| g_s2_bemf_time_cnt | int16_t | Count value of waiting for Back EMF detection. | - |
| g_s2_bemf_time_fg_calc_pre | int16_t | Count value for Electric angle 60 degrees of Back EMF detection on the last time. | - |
| g_s2_bemf_time_fg_calc_now | int16_t | Count value for Electric angle 60 degrees of Back EMF detection on this time. | - |
| g_s2_bemf_time_a_cnt | int16_t | Count value for "a" of Back EMF detection. | - |
| g_s2_bemf_time_a_cnt_pre | int16_t | Count value of "a" for Back EMF of before over a Center value of Back EMF detection. | - |
| g_s2_bemf_time_b_cnt | int16_t | Count value for "b" of Back EMF detection. | - |
| g_s2_bemf_time_b_cnt_now | int16_t | Count value of "b" for Back EMF of after over a Center value of Back EMF detection. | - |
| g_s2_bemf_time_fg_cnt | int16_t | Count value of waiting for Electric angle 30 degrees of Back EMF detection | Using for judgement of switch by Conducting pattern. |
| g_s2_bemf_fg_th_pre | int16_t | Threshold of count for waiting at Electric angle 30 degrees of Back EMF detection on the last time. | - |
| g_s2_bemf_fg_th_now | int16_t | Threshold of count for waiting at Electric angle 30 degrees of Back EMF detection on this time. | Using for judgement of switch by Conducting pattern. |
| g_s2_bemf_calc_pre | int16_t | Calculation value of Back EMF detection on the last time | - |
| g_s2_bemf_calc_now | int16_t | Calculation value of Back EMF detection on this time | - |
| g_u1_bemf_pwm_phase | uint8_t | Classification of Conducting pattern | 0: Regeneration for High side<br>1: Regeneration for Low side |
| g_s2_bemf_timeout_cnt | int16_t | Count value for time out of Back EMF detection | - |
| g_s2_bemf_time_pwmchg_cnt | int16_t | Count value by acquire of A/D conversion value for Hi-Z status detection | - |
| g_u1_bemf_chk_hiz | uint8_t | Classification of Hi-Z | 0: U phase, 1: V phase, 2: W phase |
| g_u1_bemf_chk_hiz_status | uint8_t | Management of Hi-Z status detection | 0: Hi-Z status is undetected<br>1: Hi-Z status is detected |
| g_u1_bemf_req_moten_on | uint8_t | Request of ON for Motor drive control | 0: Prohibit, 1: Approval |
| g_u1_bemf_req_moten_off | uint8_t | Request of OFF for Motor drive control | 0: Prohibit, 1: Approval |
| g_u1_bemf_mode_cnt | uint8_t | Count value of Back EMF detection mode | |
| g_u1_bemf_th_type | uint8_t | Classification of Back EMF detection | 0: Back EMF detection (Period of Regeneration for High side)<br>1: Back EMF detection (Period of Current)<br>2: Back EMF detection (Period of Regeneration for Low side) |
| g_u1_bemf_ad_type | uint8_t | Classification of selection for A/D conversion | 0: Current detection<br>1: Back EMF detection |
| g_s2_pdrv_ad_isense | int16_t | A/D conversion value of detected current | - |
| g_s2_bemf_fix_pwm_duty_cnt | int16_t | Count value of the period when PWM Duty is fixed | - |
| g_s2_bemf_timer_cnt_ave | int16_t | Electrical angle when Clamp operation started by PWM Duty | Electrical angle: Period of 360 degrees [us] |
| g_u1_bemf_limit_pwm_duty_flg | uint8_t | Clamp Flag by PWM Duty | 0: There is no Clamp Flag<br>1: There is Clamp Flag |
| g_u1_bemf_limit_vr_ad_flg | uint8_t | Level of the limit flag of the A/D value by the trigger | 0: There is no Clamp Flag<br>1: There is Clamp Flag |
| g_u2_bemf_limit_vr_ad_cnt | uint16_t | Level of the limit counter of the A/D value by the trigger | - |
| g_u1_bemf_limit_vr_ad | uint8_t | Limit value of the A/D value by the trigger | - |

## 3.4 Specification of Macro definition

Lists of macro definitions used in this sample program are shown in Table 3-7

**Table 3-7 List of Macro Definitions**

| File name | Macro name | Definition value | Remarks |
|---|---|---|---|
| control_parameter.h | CP_MAX_SPEED_RPM | 3900 | Rotation speed command maximum value (Mechanical angle) [rpm] |
| | CP_MIN_SPEED_RPM | 1200 | Rotation speed command minimum value (Mechanical angle) [rpm] |
| | CP_SPEED_LPF_K | 0.35f | LPF parameter for Rotation speed. |
| mtr_main.h | ICS_UI | 0 | Set UI to ICS |
| | BOARD_UI | 1 | Set UI to Board |
| | M_CW | 0 | User setting rotation direction: CW |
| | M_CCW | 1 | User setting rotation direction: CCW |
| | MAX_SPEED | CP_MAX_SPEED_RPM | Rotation speed command maximum value (mechanical angle) [rpm] |
| | MIN_SPEED | CP_MIN_SPEED_RPM | Rotation speed command minimum value (mechanical angle) [rpm] |
| | MARGIN_SPEED | 100 | Rotation speed command minimum value creation constants for motor stop (mechanical angle) [rpm] |
| | MARGIN_MIN_SPEED | MIN_SPEED - MARGIN_SPEED | Rotation speed command minimum value for motor stop (mechanical angle) [rpm] |
| | SPEED_LPF_K | CP_SPEED_LPF_K * 16384 | LPF parameter value for Rotation speed |
| | SW_ON | 0 | Active in case of Low |
| | SW_OFF | 1 | Active in case of High |
| | REQ_CLR | 0 | Clear Flag for stop command |
| | REQ_SET | 1 | Set Flag for stop command |
| | LED_ON_1ST_SPEED | 1500 | rotation speed LED3 ON |
| | LED_ON_2ND_SPEED | 3000 | rotation speed LED4 ON |
| | REQ_ROT_CCW | 0 | CCW: Acquisition value of Rotation direction port |
| | REQ_ROT_CW | 1 | CW: Acquisition value of Rotation direction port |
| motor_parameter.h | MP_POLE_PAIRS | 2 | Constant for correcting number of pole pairs |
| mtr_ctrl_rl78g1f_t2001.h | MTR_PWM_TIMER_FREQ | 64.0f | PWM timer count frequency [MHz] |
| | MTR_CARRIER_FREQ | 10.0f | Carrier frequency [KHz] |
| | MTR_DEADTIME | 0 | Dead Time [ns] |
| | MTR_DEADTIME_SET | MTR_DEADTIME * MTR_PWM_TIMER_FREQ / 1000 | Dead Time setting value |

| | | | |
|---|---|---|---|
| | MTR_CARRIER_SET | (MTR_PWM_TIMER_FREQ * 1000 / MTR_CARRIER_FREQ / 2) + MTR_DEADTIME_SET - 2 | Carrier setting value |
| | MTR_START_CARRIER_SET | MTR_CARRIER_SET * 30 / 100 | Carrier setting value (initial value) |
| | MTR_VR1_ADC_MAX | 802 | Trigger level A/D conversion maximum value |
| | MTR_PWM_DUTY_CALC_COEF1 | MTR_CARRIER_SET * 100 / MTR_VR1_ADC_MAX | PWM Duty calculation coefficient 1 |
| | MTR_PWM_DUTY_CALC_COEF2 | 100 | PWM Duty calculation coefficient 2 |
| | MTR_RPM_CALC_COEF1 | 556 | Target rotation speed calculation coefficient 1 |
| | MTR_RPM_CALC_COEF2 | 16888 | Target rotation speed calculation coefficient 2 |
| | MTR_RPM_CALC_COEF3 | 100 | Target rotation speed calculation coefficient 3 |
| | MTR_PORT_BEMF_ U | P5.2 | U phase port for select of phase for Measurement of Back EMF |
| | MTR_PORT_BEMF_ V | P5.3 | V phase port for select of phase for Measurement of Back EMF |
| | MTR_PORT_BEMF_ W | P5.4 | W phase port for select of phase for Measurement of Back EMF |
| | MTR_PORT_UP | P1.5 | U phase (positive phase) output port |
| | MTR_PORT_UN | P1.4 | U phase (negative phase) output port |
| | MTR_PORT_VP | P1.3 | V phase (positive phase) output port |
| | MTR_PORT_VN | P1.1 | V phase (negative phase) output port |
| | MTR_PORT_WP | P1.2 | W phase (positive phase) output port |
| | MTR_PORT_WN | P1.0 | W phase (negative phase) output port |
| | MTR_GET_ROT_DIR_REQ | P12.2 | Rotation direction detection port |
| | MTR_PORT_LED1 | P0.1 | LED1 output port |
| | MTR_PORT_LED2 | P0.0 | LED2 output port |
| | MTR_PORT_LED3 | P14.1 | LED3 output port |
| | MTR_PORT_LED4 | P14.0 | LED4 output port |
| | MTR_PORT_LED5 | P4.3 | LED5 output port |
| | MTR_PORT_LED6 | P4.2 | LED6 output port |
| | MTR_LED_ON | 0 | Active in case of Low |
| | MTR_LED_OFF | 1 | |
| | MTR_OVERVOLTAGE_LIMIT | 30 * 128 | Over voltage error determination threshold [V] |
| | MTR_UNDERVOLTAGE_LIMIT | 6 * 128 | Under voltage error determination threshold [V] |
| | MTR_VDC_SCALING | 1471 | VM voltage A/D conversion value resolution |
| | MTR_RECIVDC_SCALING | 256 | VM voltage A/D conversion value resolution (inverse) |
| | MTR_TAU1_CNT | TCR01 | TAU1 count register for rotation speed calculation |

| | | | |
|---|---|---|---|
| | MTR_ADCCH_RAJ306000_TEMP | 0 | A/D converter channel for RAJ306000 temperature measurement |
| | MTR_ADCCH_MOS_TEMP | 1 | A/D converter channel for MOS FET temperature measurement |
| | MTR_ADCCH_VM | 3 | A/D converter channel for VM voltage measurement |
| | MTR_ADCCH_BOARD_TEMP | 5 | A/D converter channel for sensor board temperature measurement |
| | MTR_ADCCH_VR1 | 6 | A/D converter channel for Trigger level |
| | MTR_ADCCH_PDRV | 7 | A/D converter channel for PreDriver voltage measurement |
| mtr_ctrl_t2001.h | MTR_LED1 | 1 | LED pattern |
| | MTR_LED2 | 2 | |
| | MTR_LED3 | 3 | |
| | MTR_LED4 | 4 | |
| | MTR_LED5 | 5 | |
| | MTR_LED6 | 6 | |
| mtr_spm_less_120_cpm.h | MTR_TWOPI | 2 * 3.14159265f | 2 Pi |
| | MTR_POLE_PAIRS | MP_POLE_PAIRS | Constant for compensate number of pole pairs |
| | MTR_RPM_RAD | 1716 | Constant to change units from [rpm] to [rad/s] |
| | MTR_RAD_RPM | 2445 / MTR_POLE_PAIRS | Constant to change units from [rad/s] to [rpm] |
| | MTR_SPEED_LIMIT_RPM | 4290 | Limit value of Rotation speed (Mechanical angle) [rpm] |
| | MTR_SPEED_LIMIT | MTR_SPEED_LIMIT_RPM / 60 * MTR_POLE_PAIRS * MTR_TWOPI * 4 | Limit value of Rotation speed (Electrical angle) [rad/s] |
| | MTR_SPEED_LPF_K | CP_SPEED_LPF_K * 16384 | LPF parameter value for Rotation speed |
| | MTR_SPEED_CALC_BASE | 383 | Constant for rotation speed measurement |
| | MTR_TIMER_CNT_BUF_NUM | 6 | Buffer size for timer count of Rotation speed measurement |
| | MTR_TIMEOUT_CNT | 200 | Waiting time for judgement of motor stop (Count value x 100[us]) |
| | MTR_START_CNT | 2000 | Boot mode period (Count value x 500[us] |
| | MTR_PATTERN_CW_V_U | 5 | CW signal pattern |
| | MTR_PATTERN_CW_W_U | 4 | |
| | MTR_PATTERN_CW_W_V | 6 | |
| | MTR_PATTERN_CW_U_V | 2 | |
| | MTR_PATTERN_CW_U_W | 3 | |
| | MTR_PATTERN_CW_V_W | 1 | |
| | MTR_PATTERN_CCW_V_U | 2 | CCW signal pattern |
| | MTR_PATTERN_CCW_V_W | 6 | |
| | MTR_PATTERN_CCW_U_W | 4 | |
| | MTR_PATTERN_CCW_U_V | 5 | |
| | MTR_PATTERN_CCW_W_V | 1 | |
| | MTR_PATTERN_CCW_W_U | 3 | |

| | | | |
|---|---|---|---|
| | MTR_PATTERN_ERROR | 0 | Conducting pattern |
| | MTR_UP_PWM_VN_ON | 1 | |
| | MTR_UP_PWM_WN_ON | 2 | |
| | MTR_VP_PWM_UN_ON | 3 | |
| | MTR_VP_PWM_WN_ON | 4 | |
| | MTR_WP_PWM_UN_ON | 5 | |
| | MTR_WP_PWM_VN_ON | 6 | |
| | MTR_UP_ON_VN_PWM | 7 | |
| | MTR_UP_ON_WN_PWM | 8 | |
| | MTR_VP_ON_UN_PWM | 9 | |
| | MTR_VP_ON_WN_PWM | 10 | |
| | MTR_WP_ON_UN_PWM | 11 | |
| | MTR_WP_ON_VN_PWM | 12 | |
| | MTR_CW | 0 | Rotation direction setting value: CW |
| | MTR_CCW | 1 | Rotation direction setting value: CCW |
| | MTR_FLG_CLR | 0 | Constant for flag clear |
| | MTR_FLG_SET | 1 | Constant for flag setting |
| | MTR_STOP_WAIT_CNT | 100 | Period to wait for motor stop<br>(Count value x 100[us]) |
| | MTR_ICS_DECIMATION | 2 | Number of function call decimation times for ICS<br>(Count value x 100[us]) |
| | MTR_SENLESS_INIT | 0 | Sensor-less mode |
| | MTR_SENLESS_INDSEN | 1 | |
| | MTR_SENLESS_BEMF | 2 | |
| | MTR_BEMF_PRE | 0 | Back EMF detection mode |
| | MTR_BEMF_1ST | 1 | |
| | MTR_BEMF_2ND | 2 | |
| | MTR_BEMF_3RD | 3 | |
| | MTR_BEMF_UP | 0 | Back EMF detection direction |
| | MTR_BEMF_DOWN | 1 | |
| | MTR_BEMF_WAIT_1ST_1 | 25 | Stop time of Motor control signal<br>(Count value x 100[us]) |
| | MTR_BEMF_WAIT_1ST_2 | 10 | Mask time of Back EMF detection<br>(Count value x 100[us]) |
| | MTR_BEMF_WAIT_1ST_3 | 0 | Mask time of Back EMF detection (Period of Current)<br>(Count value x 100[us]) |
| | MTR_BEMF_WAIT_2ND | 5 | Mask time of start for Hi-Z status detection<br>(Count value x 100[us]) |
| | MTR_BEMF_WAIT_3RD | 50 | Mask time of end for Hi-Z status detection<br>(Count value x 100[us]) |
| | MTR_BEMF_PWMCHG_WAIT | 1 | Count value for acquire of A/D value of Hi-Z status detection<br>(Count value x 100[us]) |
| | MTR_BEMF_TIMEOUT | 2500 | Time out of Back EMF detection<br>(Count value x 100[us]) |
| | MTR_BEMF_HIZ_KICK_NUM | 1 | Additional values for Current pattern of Hi-Z status detection<br>(Count value x 100[us]) |

| | | | |
|---|---|---|---|
| | MTR_BEMF_TH_LIMIT_MAX | 1024 | Maximum value of Back EMF detection value (Maximum value x 0.0049[V]) |
| | MTR_BEMF_TH_LIMIT_MIN | 0 | Minimum value of Back EMF detection value (Minimum value x 0.0049[V]) |
| | MTR_BEMF_TH_MARGIN | 300 | Limit value of Back EMF detection value (Limit value x 0.0049[V]) |
| | MTR_PWM_PHASE_P | 0 | Classification of Conducting pattern |
| | MTR_PWM_PHASE_N | 1 | |
| | MTR_BEMF_MODE_CNT | 2 | Count value of Back EMF detection mode |
| | MTR_BEMF_TIMING_TH | 50 | PWM Duty of threshold for switch of a classification for Back EMF detection [%] |
| | MTR_BEMF_FIX_PWMDUTY | 35 | Fixed value of PWM Duty [%] |
| | MTR_BEMF_FIX_PWMDUTY_CNT_LIMIT | 32 | Period when PWM Duty is fixed (Count value x Electric angle 60 degrees) |
| | MTR_BEMF_LIMIT_PWMDUTY | 55 | Clamp value of PWM Duty [%] |
| | MTR_BEMF_LIMIT_PWMDUTY_ELEROT_TIME | 1875 | Period of an electric angle of 360 degrees for start a clamp of PWM Duty [us] |
| | MTR_BEMF_LIMIT_VRAD_CNT_LIMIT | 1000 | Limit of the execution time of the A/D value by the trigger (Count value x 500[us]) |
| | MTR_BEMF_LIMIT_VRAD_COEF1 | 8 | Calculation coefficient 1 for Limit of the A/D value by the trigger |
| | MTR_BEMF_LIMIT_VRAD_COEF2 | 10 | Calculation coefficient 2 for Limit of the A/D value by the trigger |
| | MTR_WAITTIME_100_US | 100 | 100[us] wait |
| | MTR_WAITTIME_25_MS | 25000 | 25[ms] wait |
| | MTR_PATTERN_W_V | 0 | Current pattern |
| | MTR_PATTERN_W_U | 1 | |
| | MTR_PATTERN_V_U | 2 | |
| | MTR_PATTERN_V_W | 3 | |
| | MTR_PATTERN_U_W | 4 | |
| | MTR_PATTERN_U_V | 5 | |
| | MTR_PATTERN_MAX | 6 | |
| | MTR_BEMF_TH_HSIDE | 0 | Classification of Back EMF detection |
| | MTR_BEMF_TH_HIZ | 1 | |
| | MTR_BEMF_TH_LSIDE | 2 | |
| | MTR_BEMF_TH_MAX | 3 | |
| | MTR_BEMF_HIZ_U | 0 | Classification of H-Z status |
| | MTR_BEMF_HIZ_V | 1 | |
| | MTR_BEMF_HIZ_W | 2 | |
| | MTR_BEMF_HIZ_MAX | 3 | |
| | MTR_BEMF_AD_ISENSE | 0 | Classification by select of A/D channel |
| | MTR_BEMF_AD_BEMF | 1 | |
| | MTR_BEMF_AD_MAX | 2 | |
| | MTR_BEMF_SEQ_INIT | 0 | Sequence definition of Back EMF |
| | MTR_BEMF_SEQ_WAIT | 1 | |

| | | | |
|---|---|---|---|
| | MTR_BEMF_SEQ_SET_MOTEN_OFF | 2 | |
| | MTR_BEMF_SEQ_WAIT_2ND | 3 | |
| | MTR_BEMF_SEQ_CHK_HIZ | 4 | |
| | MTR_BEMF_SEQ_SET_MOTEN_ON | 5 | |
| | MTR_BEMF_SEQ_WAIT_3RD | 6 | |
| | MTR_BEMF_SEQ_CHK_TH_RANGE | 7 | |
| | MTR_BEMF_SEQ_DET_BEMF_LEVEL | 8 | |
| | MTR_BEMF_SEQ_SET_PWM | 9 | |
| | MTR_BEMF_SEQ_OVF_COUNT | 10 | |
| | MTR_BEMF_SEQ_NUM_MAX | 11 | |
| | MTR_BOOT_MODE | 0x00 | Boot mode |
| | MTR_LESS_120_MODE | 0x03 | Normal operation (LESS 120) mode |
| | MTR_ALARM_ERROR | 0x01 | ALARM error |
| | MTR_OVER_VOLTAGE_ERROR | 0x02 | Over voltage error |
| | MTR_OVER_SPEED_ERROR | 0x04 | Rotation speed abnormality error |
| | MTR_TIMEOUT_ERROR | 0x08 | Time out error |
| | MTR_LESS_ERROR | 0x10 | Current pattern error |
| | MTR_BEMF_ERROR | 0x20 | Error of Back EMF detection |
| | MTR_UNDER_VOLTAGE_ERROR | 0x40 | Under voltage error |
| | MTR_UNKNOWN_ERROR | 0x80 | Undefined error |
| | MTR_MODE_STOP | 0x00 | Stop status |
| | MTR_MODE_RUN | 0x01 | Rotating status |
| | MTR_MODE_ERROR | 0x02 | Error status |
| | MTR_SIZE_STATE | 3 | Status count |
| | MTR_EVENT_STOP | 0x00 | Motor stop event |
| | MTR_EVENT_RUN | 0x01 | Motor startup event |
| | MTR_EVENT_ERROR | 0x02 | Motor error event |
| | MTR_EVENT_RESET | 0x03 | Motor reset event |
| | MTR_SIZE_EVENT | 4 | Events count |
| | MTR_INDSEN_SEQ_STATE_END | 1 | |
| r_less_120_is.h | IS_DIRECTION_CW | MTR_CW | Direction of rotation: CW |
| | IS_DIRECTION_CCW | MTR_CCW | Direction of rotation: CCW |
| | IS_SECTION_TIME_A | 2 | Time of "a" section (x time resolution [us]) |
| | IS_SECTION_TIME_B | 4 | Time of "b" section (x time resolution [us]) |
| | IS_SECTION_TIME_C | 4 | time of "c" section (x time resolution [us]) |
| | IS_TIME_RESOLUTION | 20 | Time resolution [us] |
| | IS_PWM_OUT_TIME | 125 | Output time of motor control signal (x time resolution [us]) |
| | IS_PWM_OFF_TIME | 5 | Stop time of motor control signal (x time resolution [us]) |
| | IS_KICK_NUM | 1 | Addition value to stop position |
| | IS_TRIAL_NUM | 4 | Number of trials for inductive sense |
| | IS_STATE_INIT | 0 | Initial value |
| | IS_STATE_NORMAL | 1 | Normal |
| | IS_STATE_ERROR_PORT | 2 | Error when setting general port |
| | IS_STATE_ERROR_WAIT | 3 | Error when waiting for a specified time |

| | | | |
|---|---|---|---|
| | IS_STATE_ERROR_WDT | 4 | Error when clearing watchdog timer |
| | IS_STATE_ERROR_ADC | 5 | Error when getting A/D conversion result |
| | IS_STATE_MAX | 6 | Maximum value |
| | IS_PATTERN_1 | 0 | Current pattern 1 |
| | IS_PATTERN_2 | 1 | Current pattern 2 |
| | IS_PATTERN_3 | 2 | Current pattern 3 |
| | IS_PATTERN_4 | 3 | Current pattern 4 |
| | IS_PATTERN_5 | 4 | Current pattern 5 |
| | IS_PATTERN_6 | 5 | Current pattern 6 |
| | IS_PATTERN_MAX | 6 | Maximum value |
| r_less_120_isw.h | IS_PORT_OFF | 0x00U | General port setting: all OFF |
| | IS_PORT_P_ON | 0x2CU | General port setting: $U_p$, $V_p$, $W_p$ = ON |
| | IS_PORT_N_ON | 0x13U | General port setting: $U_n$, $V_n$, $W_n$ = ON |
| | ISW_STATE_INIT | 0 | Initial value |
| | ISW_STATE_NORMAL | 1 | Normal |
| | ISW_STATE_ERROR | 2 | Error |
| | ISW_STATE_MAX | 3 | Maximum value |
| version.h | FW_REVISION | 103 | F/W Revision information |

| File Name | Macro Name | Content | Remark |
|---|---|---|---|
| r_cg_userdefine.h | SPI00_CS_H | P0 = P0 \| 0x20 | SPI communication Chip Select signal = H |
| | SPI00_CS_L | P0 = P0 & 0xDF | SPI communication Chip Select signal = L |
| | SPI_WAIT_MODE | 0x01 | SPI communication Wait mode |
| | SPI_INTR_MODE | 0x02 | SPI communication Interrupt mode |
| | SPI_ADC_MODE | 0x03 | SPI communication ADC mode |

| File Name | Macro Name | Content | Remarks |
|---|---|---|---|
| r_cg_predrv.h | REG_BUFF_SIZE | 2 | PreDriver register buffer size |
| | SPI_CHK_MAX | 100 | PreDriver SPI communication check count |
| | PREDRV_NORMAL | 0 | PreDriver sequence none |
| | PREDRV_SPI_ERROR | 1 | PreDriver sequence SPI communication error |
| | PREDRV_ALARM_ERROR | 2 | PreDriver sequence ALARM error |
| | PREDRV_REGRW_ERROR | 4 | PreDriver sequence Register R/W error |
| | PREDRV_SPI_ACCESS_OK | 0x6A | PreDriver SPI communication judgement |
| | PREDRV_ALMRAW1_OK | 0xEF | PreDriver ALMRAW1 judgement |
| | HALL_SIG_MASK | 0x7F | HALL_SIG mask |
| | ALMSTS1_TSD_N | 0x01 | ALARM Status1 judgement |
| | ALMSTS1_OCP_N | 0x02 | |
| | ALMSTS1_VGB_UVP_N | 0x04 | |
| | ALMSTS1_VGB_OVP_N | 0x08 | |
| | ALMSTS1_VGT_UVP_N | 0x10 | |
| | ALMSTS1_VGT_OVP2_N | 0x20 | |
| | ALMSTS1_VGT_OVP1_N | 0x40 | |
| | ALMSTS1_VREG5_OVP_N | 0x80 | |
| | ALMSTS1_NO_ERROR | 0xEF | |
| | ALMSTS1_VGT_UVP_MASK | 0xEF | |
| | ALMSTS2_VM_UVP_N | 0x01 | ALARM Status2 judgement |
| | ALMSTS2_DI_SEL_W_CMP_N | 0x20 | |
| | ALMSTS2_DI_SEL_V_CMP_N | 0x40 | |
| | ALMSTS2_DI_SEL_U_CMP_N | 0x80 | |
| | ALMSTS2_NO_ERROR | 0xFF | |
| | WHO_AM_I_MASK | 0xFE | WHO_AM_I mask |
| | INIT_PS_ALL | 0x01 | PS_ALL initial value |
| | INIT_PS_1ST | 0x3E | PS initial value 1st |
| | INIT_PS_2ND | 0x3F | PS initial value 2nd |
| | INIT_PS_3RD | 0xBF | PS initial value 3rd |
| | INIT_SELSIG_U | 0x03 | SELSIG_U initial value |
| | INIT_SELSIG_V | 0x14 | SELSIG_V initial value |
| | INIT_SELSIG_W | 0x25 | SELSIG_W initial value |
| | INIT_HALL_SIG | 0xA0 | Initial value of HALL_SIG |
| | INIT_ALMOPE1 | 0x10 | ALMOPE1 initial value |
| | INIT_ALMOUT1 | 0x10 | ALMOUT1 initial value |

| | | | |
|---|---|---|---|
| | INIT_CS_SET2 | 0x60 | CS_SET2 initial value |
| | INIT_ERROR_WAIT | 0x00 | ERROR_WAIT initial value |
| | INIT_CS_SET1 | 0x08 | CS_SET1 initial value |
| | INIT_HAIC_TH | 0x00 | HAIC_TH initial value |
| | INIT_LD_WAIT | 0x00 | LD_WAIT initial value |
| | INIT_DRIVE_SET | 0x01 | DRIVE_SET initial value |
| | INIT_IDRCNT_H | 0x00 | IDRCNT_H initial value |
| | INIT_IDRCNT_L | 0x00 | IDRCNT_L initial value |
| | INIT_TRCNT_P | 0x00 | TRCNT_P initial value |
| | INIT_CPSET1 | 0x01 | CPSET1 initial value |
| | INIT_CPSET2 | 0x02 | CPSET2 initial value |
| | INIT_CP_TRIM | 0x00 | CP_TRIM initial value |
| | INIT_VREG5_TRIM | 0x20 | VREG5_TRIM initial value |
| | INIT_CSAMP_TRIM | 0x20 | CSAMP_TRIM initial value |
| | INIT_TRIM_PT | 0x00 | TRIM_PT initial value protected |
| | INIT_TRIM_PT_UP | 0x95 | TRIM_PT initial value unprotected |
| | INIT_TRIM_EN | 0x00 | TRIM_EN initial value |
| | INIT_TRIM_EN_EFWD | 0x01 | TRIM_EN initial value valid trimming data |
| | INIT_BGR_TRIM | 0x00 | BGR_TRIM initial value |
| | INIT_BFAMP_TRIM | 0x00 | BFAMP_TRIM initial value |
| | ERRRCV_PS_1ST | 0x3C | PS ALARM recovery value1st |
| | ERRRCV_PS_2ND | 0x3E | PS ALARM recovery value 2nd |
| | ERRRCV_PS_3RD | 0x3F | PS ALARM recovery value 3rd |
| | ERRRCV_PS_4TH | 0xBF | PS ALARM recovery value 4th |
| | ERRRCV_MOT_EN_CLR | 0x00 | DRIVE_SET ALARM recovery value Prohibition of motor rotation |
| | ERRRCV_MOT_EN_SET | 0x01 | DRIVE_SET ALARM recovery value Permission of motor rotation |
| | ERRRCV_ALM_LATCH_CLR | 0x40 | DRIVE_SET ALARM recovery value Clear of ALARM latch |
| | WAITTIME_1_MS | 0x11F8 | 1[ms] wait |
| | WAITTIME_3_MS | 0x35E8 | 3[ms] wait |
| | REQ_MOT_EN_OFF | 0x00 | Motor drive control: OFF |

| | | | |
|---|---|---|---|
| | REQ_MOT_EN_ON | 0x01 | Motor drive control: ON |
| | SET_MOT_EN_CLR | 0x00 | DRIVE_SET: Inhibit of Motor rotation |
| | SET_MOT_EN_SET | 0x01 | DRIVE_SET: Permission of Motor rotation |
| | REQ_ADC_SEL_ISEN | 0x00 | Indicate for the current setting |
| | REQ_ADC_SEL_BEMF | 0x01 | Indicate for the Back EMF setting |
| | SET_ADC_SEL_ISEN | 0x01 | Current setting of ADC_SEL |
| | SET_ADC_SEL_BEMF | 0x03 | Back EMF of ADC_SEL |
| | INIT_ICS_PS_ALL | INIT_PS_ALL | PS_ALL ICS variable initial value |
| | INIT_ICS_PS | INIT_PS_3RD | PS ICS variable initial value |
| | INIT_ICS_SW_RESET | 0x00 | SW_RESET ICS variable initial value |
| | INIT_ICS_ADC_SEL | 0x00 | ADC_SEL ICS variable initial value |
| | INIT_ICS_SELSIG_U | 0x03 | SELSIG_U ICS variable initial value |
| | INIT_ICS_SELSIG_V | 0x14 | SELSIG_V ICS variable initial value |
| | INIT_ICS_SELSIG_W | 0x25 | SELSIG_W ICS variable initial value |
| | INIT_ICS_HALL_SIG | INIT_HALL_SIG | HALL_SIG ICS variable initial value |
| | INIT_ICS_ALMSTS1 | 0xFF | ALMSTS1 ICS variable initial value |
| | INIT_ICS_ALMOPE1 | INIT_ALMOPE1 | ALMOPE1 ICS variable initial value |
| | INIT_ICS_ALMOUT1 | INIT_ALMOUT1 | ALMOUT1 ICS variable initial value |
| | INIT_ICS_ALMSTS2 | 0xFF | ALMSTS2 ICS variable initial value |
| | INIT_ICS_CS_SET2 | INIT_CS_SET2 | CS_SET2 ICS variable initial value |
| | INIT_ICS_ALMOUT2 | 0x00 | ALMOUT2 ICS variable initial value |
| | INIT_ICS_ERROR_WAIT | 0x00 | ERROR_WAIT ICS variable initial value |
| | INIT_ICS_CS_SET1 | INIT_CS_SET1 | CS_SET1 ICS variable initial value |
| | INIT_ICS_HAIC_TH | 0x00 | HAIC_TH ICS variable initial value |
| | INIT_ICS_PDDSTS | 0xF0 | PDDSTS ICS variable initial value |
| | INIT_ICS_LD_WAIT | 0x00 | LD_WAIT ICS variable initial value |

| | | | |
|---|---|---|---|
| | INIT_ICS_DRIVE_SET | INIT_DRIVE_SET | DRIVE_SET ICS variable initial value |
| | INIT_ICS_DI_TIME | 0x00 | DI_TIME ICS variable initial value |
| | INIT_ICS_IDRCNT_H | 0x00 | IDRCNT_H ICS variable initial value |
| | INIT_ICS_IDRCNT_L | 0x00 | IDRCNT_L ICS variable initial value |
| | INIT_ICS_TRCNT_P | 0x00 | TRCNT_P ICS variable initial value |
| | INIT_ICS_CPSET1 | 0x01 | CPSET1 ICS variable initial value |
| | INIT_ICS_CPSET2 | 0x02 | CPSET2 ICS variable initial value |
| | INIT_ICS_CP_TRIM | INIT_CP_TRIM | CP_TRIM ICS variable initial value |
| | INIT_ICS_VREG5_TRIM | INIT_VREG5_TRIM | VREG5_TRIM ICS variable initial value |
| | INIT_ICS_CSAMP_TRIM | INIT_CSAMP_TRIM | CSAMP_TRIM ICS variable initial value |
| | INIT_ICS_ALMRAW1 | 0xFF | ALMRAW1 ICS variable initial value |
| | INIT_ICS_TOIN_MONI | 0x00 | TOIN_MONI ICS variable initial value |
| | INIT_ICS_WHO_AM_I | 0x6A | WHO_AM_I ICS variable initial value |
| | INIT_ICS_TRIM_PT | INIT_TRIM_PT | TRIM_PT ICS variable initial value |
| | INIT_ICS_TRIM_EN | INIT_TRIM_EN | TRIM_EN ICS variable initial value |
| | INIT_ICS_BGR_TRIM | INIT_BGR_TRIM | BGR_TRIM ICS variable initial value |
| | INIT_ICS_BFAMP_TRIM | INIT_BFAMP_TRIM | BFAMP_TRIM ICS variable initial value |
| | SEQ_INIT | 0 | PreDriver initial sequence definition |
| | SEQ_CHK_SPI | 1 | |
| | SEQ_CHK_TSD_N | 2 | |
| | SEQ_SET_HALL_SIG_PRM | 3 | |
| | SEQ_SET_5VTRIM | 4 | |
| | SEQ_SET_ALMOPE1_PRM | 5 | |
| | SEQ_SET_ALMOUT1_PRM | 6 | |
| | SEQ_SET_CS_SET2_PRM | 7 | |
| | SEQ_SET_CS_SET1_PRM | 8 | |
| | SEQ_SET_SEQINIT_PRM | 9 | |
| | SEQ_SET_PS_ALL_PRM | 10 | |
| | SEQ_SET_PS_1ST_PRM | 11 | |
| | SEQ_SET_PS_2ND_PRM | 12 | |
| | SEQ_CHK_ALMRAW1 | 13 | |
| | SEQ_SET_PS_3RD_PRM | 14 | |
| | SEQ_CHK_ALMSTS | 15 | |
| | SEQ_SET_MOT_EN | 16 | |
| | SEQ_END | 17 | |
| | SEQ_NUM_MAX | 18 | |

| | | | |
|---|---|---|---|
| | ERR_RCV_SEQ_INIT | 0 | PreDriver ALARM recovery sequence definition |
| | ERR_RCV_SEQ_CHK_STS | 1 | |
| | ERR_RCV_SEQ_CLR_MOT_EN | 2 | |
| | ERR_RCV_SEQ_SET_PS_1ST | 3 | |
| | ERR_RCV_SEQ_CHK_ALMSTS_1ST | 4 | |
| | ERR_RCV_SEQ_SET_ALM_LATCH_CLR | 5 | |
| | ERR_RCV_SEQ_SET_PS_2ND | 6 | |
| | ERR_RCV_SEQ_SET_PS_3RD | 7 | |
| | ERR_RCV_SEQ_CHK_ALMRAW1 | 8 | |
| | ERR_RCV_SEQ_SET_PS_4TH | 9 | |
| | ERR_RCV_SEQ_CHK_ALMSTS_2ND | 10 | |
| | ERR_RCV_SEQ_SET_MOT_EN | 11 | |
| | ERR_RCV_SEQ_END | 12 | |
| | ERR_RCV_SEQ_NUM_MAX | 13 | |

## 3.5 Flow chart

Figure 3-8 shows the whole flow chart and the flow chart of initialization function.

Also, flow chart of main processing in sample program are shown in Figure 3-8 to Figure 3-18.

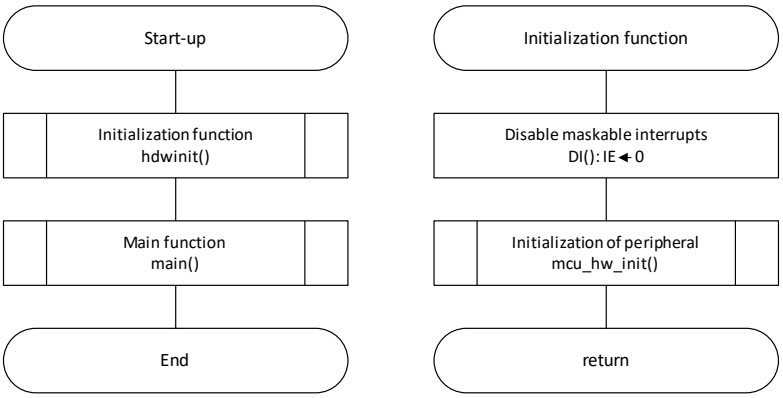


**Figure 3-8 Flow chart (Overall and Initialization function)**

### 3.5.1 **Main function**

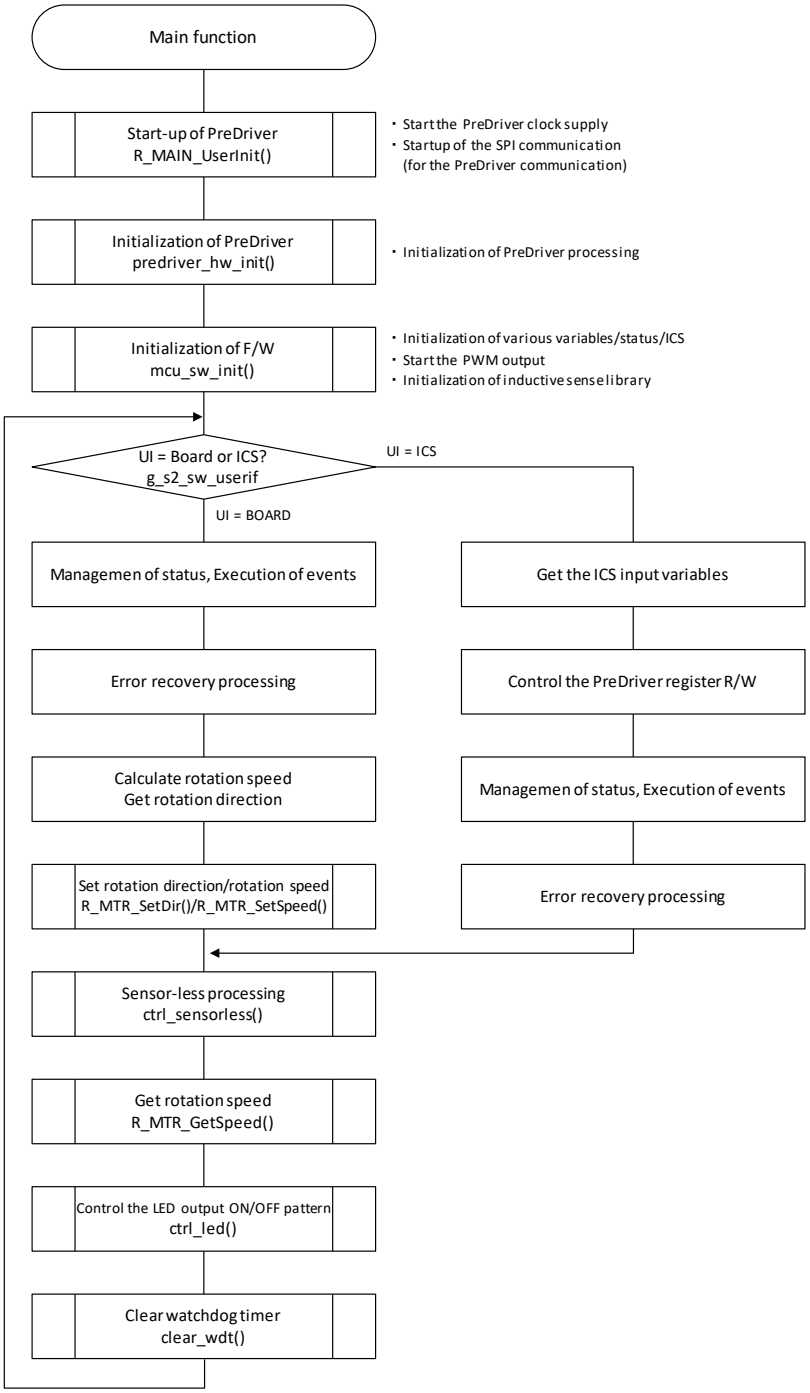


**Figure 3-9 Flow chart (Main function)**

### 3.5.2 Initialization of PreDriver processing

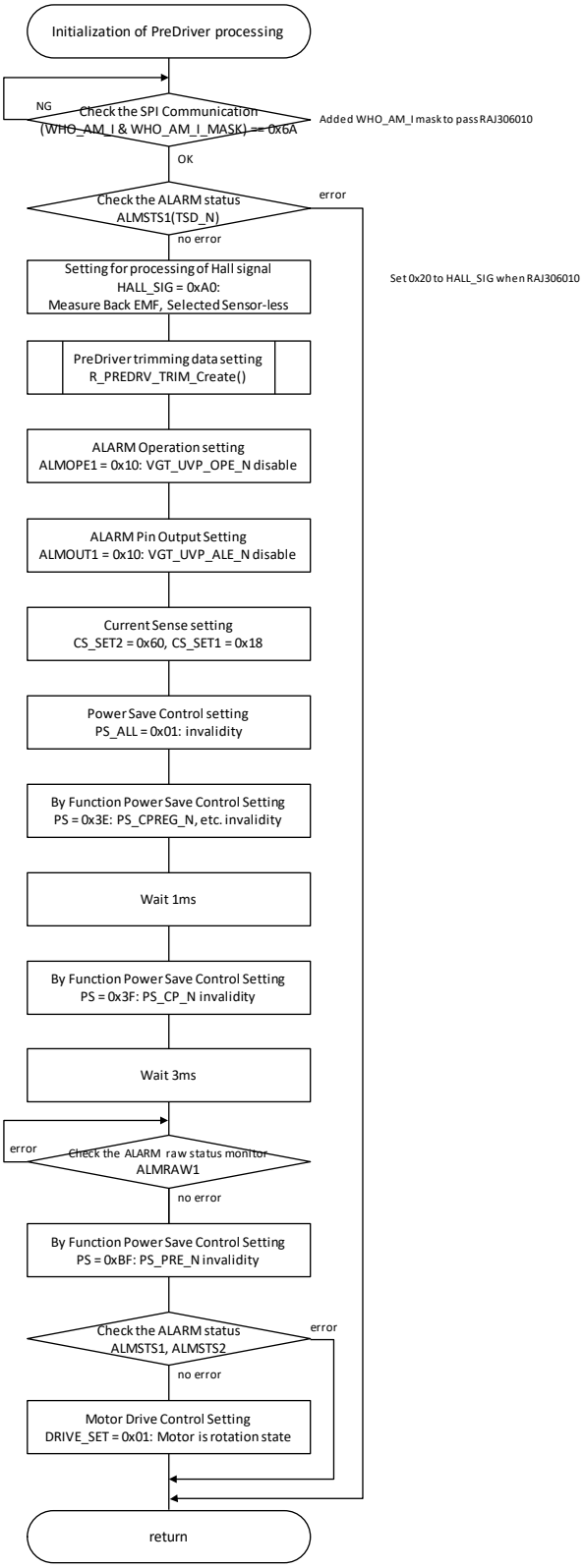


**Figure 3-10 Flow chart (Initialization of PreDriver processing)**

### 3.5.3 Sensor-less processing: Inductive sense processing

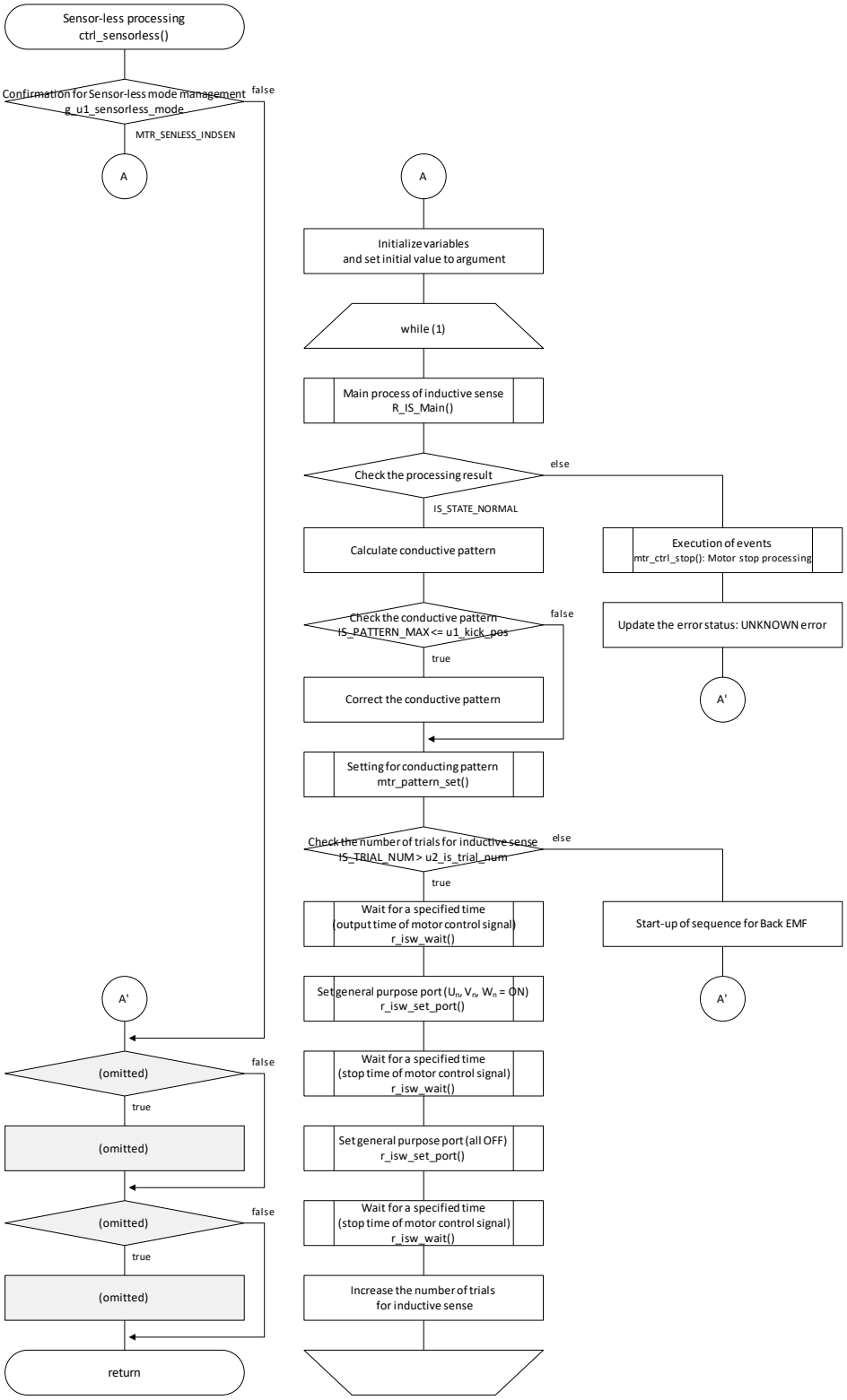


**Figure 3-11 Flow chart (Sensor-less processing: Inductive sense processing)**

### 3.5.4 Carrier frequency interruption processing


**Figure 3-12 Flow chart (Carrier frequency interruption processing)**

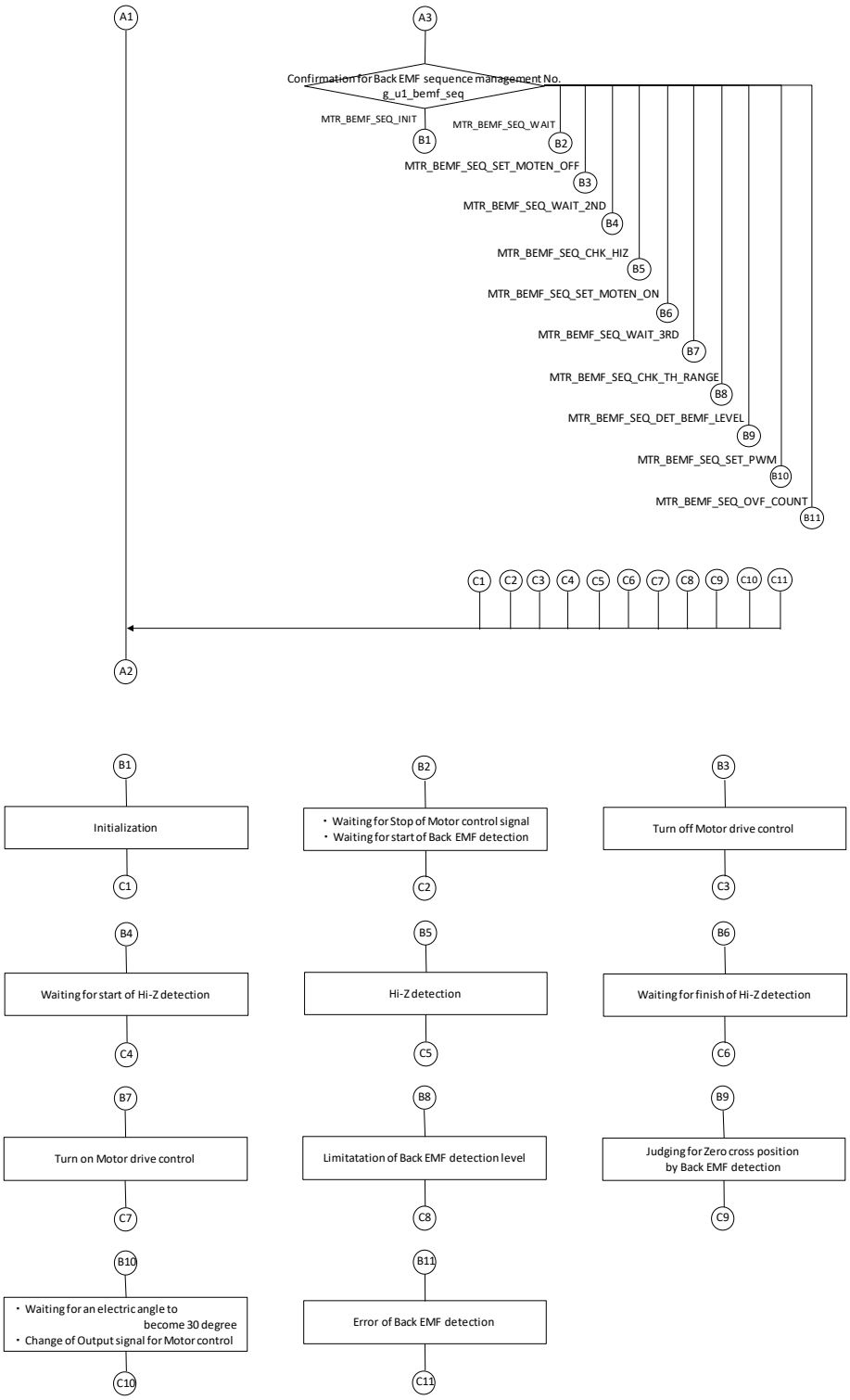


**Figure 3-13 Flow chart (Carrier frequency interruption processing)**

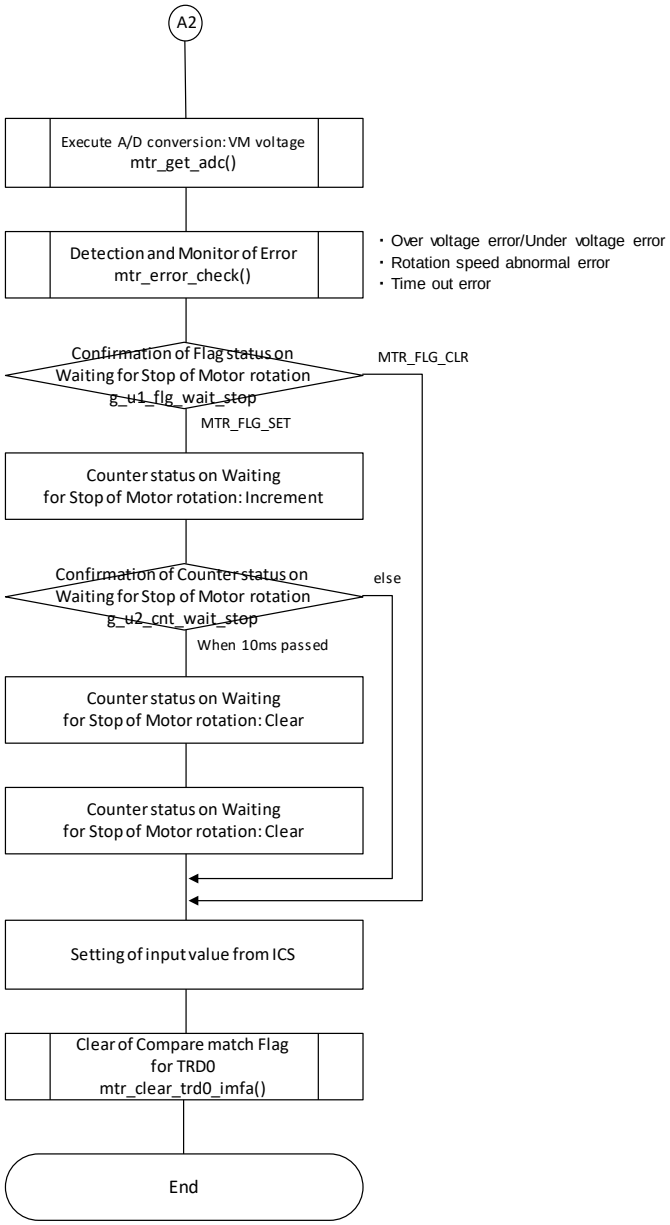


**Figure 3-14 Flow chart (Carrier frequency interruption processing)**

### 3.5.5 Underflow of Carrier frequency Interruption processing

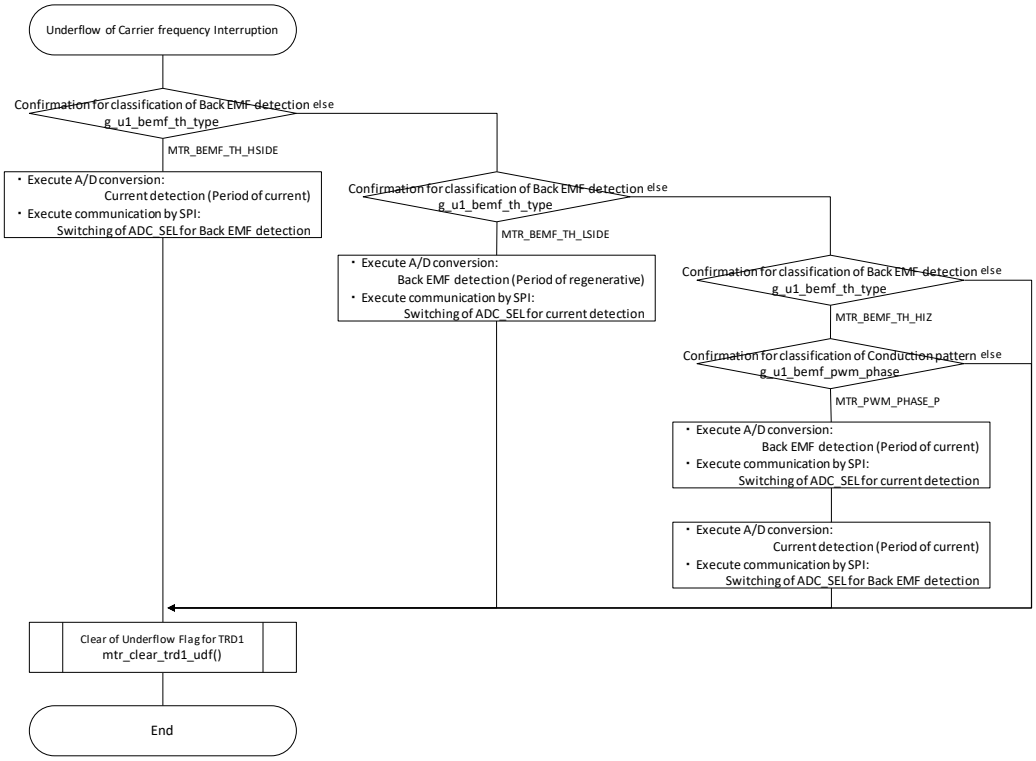


**Figure 3-15 Flow chart (Underflow of Carrier frequency Interruption processing)**

### 3.5.6 **500[us] interruption processing**

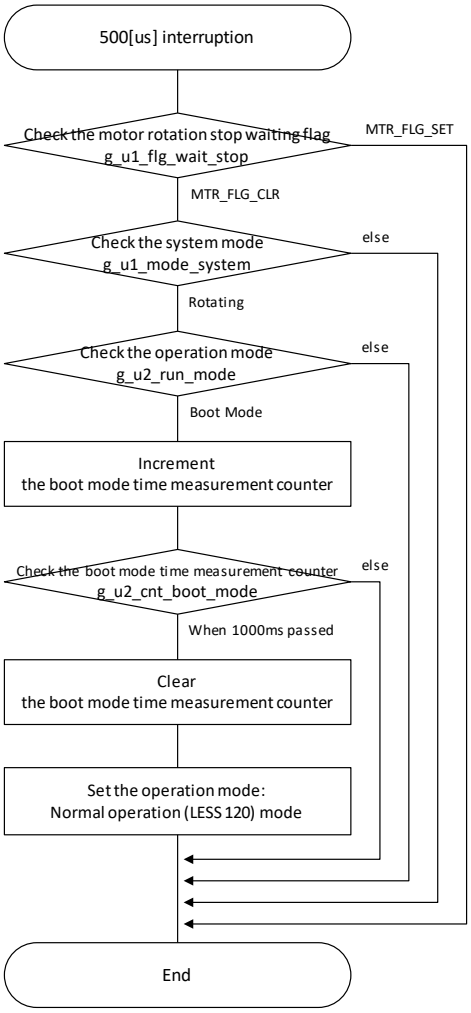


**Figure 3-16 Flow chart (500[us] interruption processing)**

### 3.5.7 ALARM interruption processing

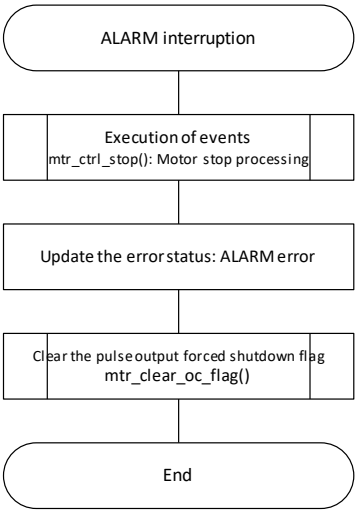


**Figure 3-17 Flow chart (ALARM interruption processing)**

### 3.5.8 ALARM recovery processing

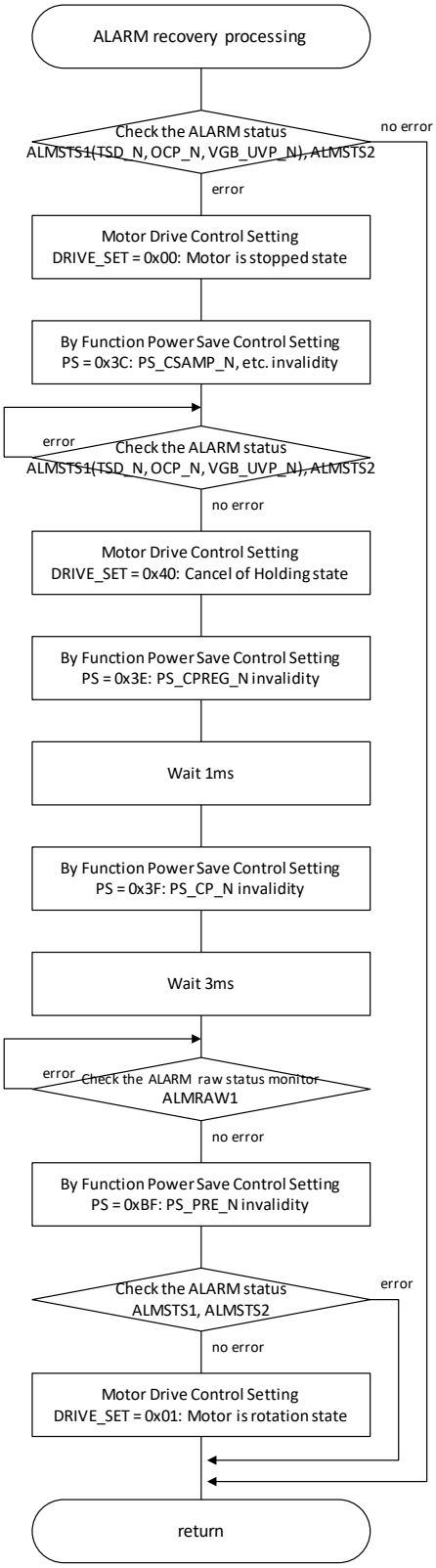


**Figure 3-18 Flow chart (ALARM recovery processing)**

# 4. Development support tool In Circuit Scope

## 4.1 Overview

In the target sample programs described in this application note, user interfaces (rotating/stop command, rotation speed command, etc.) based on the development support tool 'In Circuit Scope' (ICS) can be used. ICS is a tool which displays on PC real-time waveforms of global variables of the program being executed on the target system. Refer to 'In Circuit Scope manual' for usage and more details.

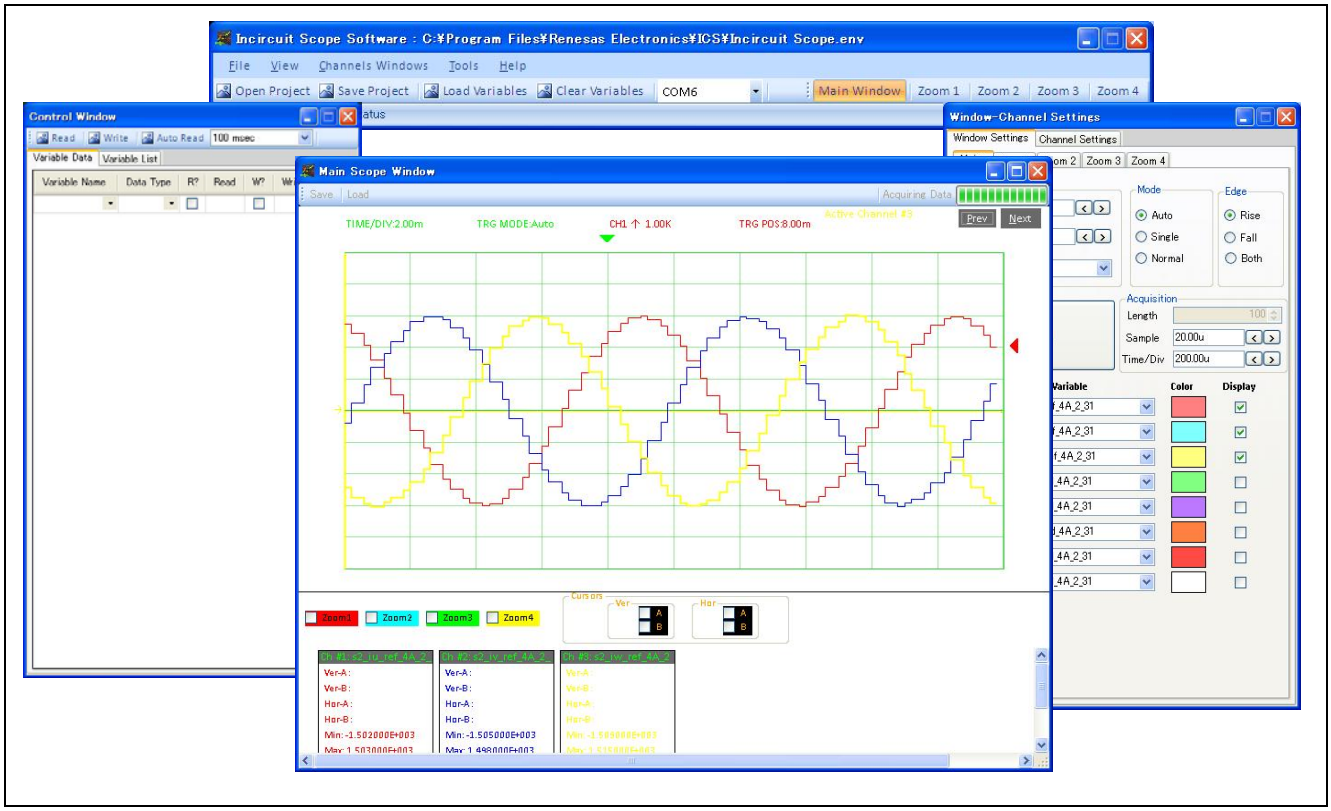


**Figure 4-1 In Circuit Scope - Appearance**

## 4.2 How to use library

In order to use ICS, it is necessary to call functions related to ICS. The ICS-related functions have been set by conditional compilation (#ifdef--#endif). To use ICS, set as follows.
[File name] mtr_common.h
[Point to change] Add the following declaration.
#define ICS_USE

## 4.3 List of variables for ICS

Table 4-1 and Table 4-2 are list of variables for ICS. Table 4-1 variable values are reflected to the protect variables when the same values as g_s2_enable_write are written to com_s2_enable_write.
Table 4-2 variable values do not depend on com_s2_enable_write.

**Table 4-1 List of Variables for ICS**

| Variable bame | Type | Content | Remarks ([ ]: protect variable name) |
|---|---|---|---|
| com_s2_direction | int16_t | Rotation direction<br>0: CW<br>1: CCW | [g_u1_direction] |
| com_s2_ref_speed_rpm | int16_t | Rotation speed command value (mechanical angle) [rpm]<br>* Not Used | [g_s2_ref_speed] |
| com_s2_speed_lpf_k | int16_t | speed LPF parameter | [g_s2_speed_lpf_k] |
| com_s2_enable_write | int16_t | Enable to rewriting variables | - |

**Table 4-2 List of Variables for ICS**

| Variable name | Type | Content | Remarks |
|---|---|---|---|
| com_s2_sw_userif | int16_t | User interface switch<br>0: ICS user interface use<br>1: Board user interface use | - |
| com_s2_mode_system | int16_t | State management<br>0: Stop mode<br>1: Run mode<br>3: Reset | - |
| com_s2_pwm_duty | int16_t | Compare register setting value of Timer RD | PWM Duty setting value: 0 to 3198<br>PWM Duty [%]: 0 to 100 |
| com_u1_pdrvreg_ctrl | uint8_t | PreDriver register R/W control flag | 0: R/W disable<br>1: R/W enable |
| com_u1_pdrvreg_ps_all_pre | uint8_t | PreDriver register PS_ALL previous value | Read value |
| com_u1_pdrvreg_ps_all_now | uint8_t | PreDriver register PS_ALL current value | Write value |
| com_u1_pdrvreg_ps_pre | uint8_t | PreDriver register PS previous value | Read value |
| com_u1_pdrvreg_ps_now | uint8_t | PreDriver register PS current value | Write value |
| com_u1_pdrvreg_sw_reset_pre | uint8_t | PreDriver register SW_RESET previous value | Read value |
| com_u1_pdrvreg_sw_reset_now | uint8_t | PreDriver register SW_RESET current value | Write value |
| com_u1_pdrvreg_adc_sel_pre | uint8_t | PreDriver register ADC_SEL previous value | Read value |
| com_u1_pdrvreg_adc_sel_now | uint8_t | PreDriver register ADC_SEL current value | Write value |
| com_u1_pdrvreg_selsig_u_pre | uint8_t | PreDriver register SELSIG_U previous value | Read value |
| com_u1_pdrvreg_selsig_u_now | uint8_t | PreDriver register SELSIG_U current value | Write value |
| com_u1_pdrvreg_selsig_v_pre | uint8_t | PreDriver register SELSIG_V previous value | Read value |
| com_u1_pdrvreg_selsig_v_now | uint8_t | PreDriver register SELSIG_V current value | Write value |
| com_u1_pdrvreg_selsig_w_pre | uint8_t | PreDriver register SELSIG_W previous value | Read value |
| com_u1_pdrvreg_selsig_w_now | uint8_t | PreDriver register SELSIG_W current value | Write value |
| com_u1_pdrvreg_hall_sig_pre | uint8_t | PreDriver register HALL_SIG previous value | Read value |
| com_u1_pdrvreg_hall_sig_now | uint8_t | PreDriver register HALL_SIG current value | Write value |
| com_u1_pdrvreg_almsts1_pre | uint8_t | PreDriver register ALMSTS1 previous value | Read value (ALMSTS1 Read Only) |
| com_u1_pdrvreg_almope1_pre | uint8_t | PreDriver register ALMOPE1 previous value | Read value |

| | | | |
|---|---|---|---|
| com_u1_pdrvreg_almope1_now | uint8_t | PreDriver register ALMOPE1 current value | Write value |
| com_u1_pdrvreg_almout1_pre | uint8_t | PreDriver register ALMOUT1 previous value | Read value |
| com_u1_pdrvreg_almout1_now | uint8_t | PreDriver register ALMOUT1 current value | Write value |
| com_u1_pdrvreg_almsts2_pre | uint8_t | PreDriver register ALMSTS2 previous value | Read value (ALMSTS2 Read Only) |
| com_u1_pdrvreg_cs_set2_pre | uint8_t | PreDriver register CS_SET2 previous value | Read value |
| com_u1_pdrvreg_cs_set2_now | uint8_t | PreDriver register CS_SET2 current value | Write value |
| com_u1_pdrvreg_almout2_pre | uint8_t | PreDriver register ALMOUT2 previous value | Read value |
| com_u1_pdrvreg_almout2_now | uint8_t | PreDriver register ALMOUT2 current value | Write value |
| com_u1_pdrvreg_error_wait_pre | uint8_t | PreDriver register ERROR_WAIT previous value | Read value |
| com_u1_pdrvreg_error_wait_now | uint8_t | PreDriver register ERROR_WAIT current value | Write value |
| com_u1_pdrvreg_cs_set1_pre | uint8_t | PreDriver register CS_SET1 previous value | Read value |
| com_u1_pdrvreg_cs_set1_now | uint8_t | PreDriver register CS_SET1 current value | Write value |
| com_u1_pdrvreg_haic_th_pre | uint8_t | PreDriver register HAIC_TH previous value | Read value |
| com_u1_pdrvreg_haic_th_now | uint8_t | PreDriver register HAIC_TH current value | Write value |
| com_u1_pdrvreg_pddsts_pre | uint8_t | PreDriver register PDDSTS previous value | Read value (PDDSTS Read Only) |
| com_u1_pdrvreg_ld_wait_pre | uint8_t | PreDriver register LD_WAIT previous value | Read value |
| com_u1_pdrvreg_ld_wait_now | uint8_t | PreDriver register LD_WAIT current value | Write value |
| com_u1_pdrvreg_drive_set_pre | uint8_t | PreDriver register DRIVE_SET previous value | Read value |
| com_u1_pdrvreg_drive_set_now | uint8_t | PreDriver register DRIVE_SET current value | Write value |
| com_u1_pdrvreg_di_time_pre | uint8_t | PreDriver register DI_TIME previous value | Read value |
| com_u1_pdrvreg_di_time_now | uint8_t | PreDriver register DI_TIME current value | Write value |
| com_u1_pdrvreg_idrcnt_h_pre | uint8_t | PreDriver register IDRCNT_H previous value | Read value |
| com_u1_pdrvreg_idrcnt_h_now | uint8_t | PreDriver register IDRCNT_H current value | Write value |
| com_u1_pdrvreg_idrcnt_l_pre | uint8_t | PreDriver register IDRCNT_L previous value | Read value |
| com_u1_pdrvreg_idrcnt_l_now | uint8_t | PreDriver register IDRCNT_L current value | Write value |
| com_u1_pdrvreg_trcnt_p_pre | uint8_t | PreDriver register TRCNT_P previous value | Read value |
| com_u1_pdrvreg_trcnt_p_now | uint8_t | PreDriver register TRCNT_P current value | Write value |
| com_u1_pdrvreg_cpset1_pre | uint8_t | PreDriver register CPSET1 previous value | Read value |
| com_u1_pdrvreg_cpset1_now | uint8_t | PreDriver register CPSET1 current value | Write value |
| com_u1_pdrvreg_cpset2_pre | uint8_t | PreDriver register CPSET2 previous value | Read value |
| com_u1_pdrvreg_cpset2_now | uint8_t | PreDriver register CPSET2 current value | Write value |
| com_u1_pdrvreg_cp_trim_pre | uint8_t | PreDriver register CP_TRIM previous value | Read value |
| com_u1_pdrvreg_cp_trim_now | uint8_t | PreDriver register CP_TRIM current value | Write value |
| com_u1_pdrvreg_vreg5_trim_pre | uint8_t | PreDriver register VREG5_TRIM previous value | Read value |
| com_u1_pdrvreg_vreg5_trim_now | uint8_t | PreDriver register VREG5_TRIM current value | Write value |
| com_u1_pdrvreg_csamp_trim_pre | uint8_t | PreDriver register CSAMP_TRIM previous value | Read value |
| com_u1_pdrvreg_csamp_trim_now | uint8_t | PreDriver register CSAMP_TRIM current value | Write value |
| com_u1_pdrvreg_almraw1_pre | uint8_t | PreDriver register ALMRAW1 previous value | Read value (ALMRAW1 Read Only) |
| com_u1_pdrvreg_toin_moni_pre | uint8_t | PreDriver register TOIN_MONI previous value | Read value (TOIN_MONI Read Only) |
| com_u1_pdrvreg_who_am_i_pre | uint8_t | PreDriver register WHO_AM_I previous value | Read value (WHO_AM_I Read Only) |
| com_u1_pdrvreg_trim_pt_pre | uint8_t | PreDriver register TRIM_PT previous value | Read value |
| com_u1_pdrvreg_trim_pt_now | uint8_t | PreDriver register TRIM_PT current value | Write value |
| com_u1_pdrvreg_trim_en_pre | uint8_t | PreDriver register TRIM_EN previous value | Read value |
| com_u1_pdrvreg_trim_en_now | uint8_t | PreDriver register TRIM_EN current value | Write value |
| com_u1_pdrvreg_bgr_trim_pre | uint8_t | PreDriver register BGR_TRIM previous value | Read value |
| com_u1_pdrvreg_bgr_trim_now | uint8_t | PreDriver register BGR_TRIM current value | Write value |
| com_u1_pdrvreg_bfamp_trim_pre | uint8_t | PreDriver register BFAMP_TRIM previous value | Read value |
| com_u1_pdrvreg_bfamp_trim_now | uint8_t | PreDriver register BFAMP_TRIM current value | Write value |

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Dec.22.17 | - | First edition issued. |
| 1.01 | Feb.28.19 | - | RAJ306000_LESS_120_OPEN_V101<br>Corrected the timing of the interval timer from 5[us] to 20[us].<br>Change parameters for RSSK (TG-55L-KA).<br>Correction of errors.<br>Figure 2-5, 3-4, 5, 9, 12, 13, 14<br>Table 3-6: g_u2_fw_revision, g_u2_lib_revision<br>Table 3-7: SPI_INTR_MODE, POLE_PAIR, etc. |
| 1.02 | Jun.05.19 | - | RAJ306000_LESS_120_OPEN_*_*_V102<br>Support IDE: CS+ for CC, e$^2$ studio<br>Table 1-1, 2-10, 11<br>Change library for ICS.<br>Table 2-10: ics<br>Update inductive sense library.<br>Figure 3-11, Table 2-10, 12, 3-4, 6 ,7<br>Remove 20[us] interruption processing.<br>Figure 3-11, Table 2-4, 9, 3-4<br>Correction of errors.<br>Figure 3-12, Table 3-6, etc. |
| 1.03 | Apr.08.20 | - | RAJ306000_LESS_120_OPEN_*_*_V103<br>Added macro definition (HALL_SIG_MASK, WHO_AM_I_MASK)<br>Figure 3-10, Table 3-7<br>Correction of errors.<br>Table 4-2: TRIM_PT, TRIM_EN, BGR_TRIM, BFAMP_TRIM, etc. |
| | | | |

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

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

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

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

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

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

# Notice

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

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

(Rev.4.0-1 November 2017)

## Corporate Headquarters

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

## Trademarks

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

## Contact information

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