Summary

This application note aims at explaining the sample program for operating the 3 phase brushless DC motor with sensorless 120 degrees conducting method, by using the functions of RL78/F14.

Sample program is only to be used as reference and Renesas Electronics Corporation does not guarantee the operations. Before sample program, carry out a thorough evaluation in a suitable environment.

Operation checking device

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

- RL78/F14 (R5F10PLJ)

Contents

1. Overview .......................................................... 2
2. System overview ................................................. 3
3. Motor control method ........................................... 7
4. Description of peripheral functions used .................... 16
5. Description of control program ............................... 21
1. Overview

This application note describes an example of speed control by sensorless 120 degrees conducting method of brushless DC motor (herein after referred to as BLDC motor) by using micro controller RL78/F14.

1.1 Usage of the system

This system (sample program) enables 120 degrees conducting control by using an RL78/F14 micro controller mounted CPU board, an inverter board for motor control (ECU001-F14-12V note 1) and a BLDC motor (BLY171S-15V-8000 note 2).

Notes:  1. Evaluation board “ECU001-F14-12V” is products of Desk Top Lab Inc.  
       (http://www.desktoplab.co.jp/)
2. BLDC Motor “BLY171S-15V-8000” is products of Anaheim Automation Inc..  
       (http://www.anaheimautomation.com/)

1.2 Development environment

(1) Software development environment

<table>
<thead>
<tr>
<th>Integrated development environment</th>
<th>CS+ for CA, CX (V4.00.00)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CS+ for CC (V5.00.00)</td>
</tr>
<tr>
<td></td>
<td>IAR Embedded Workbench (Ver. 7.4.1.4269)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Build tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA78K0R (V1.72)</td>
</tr>
<tr>
<td>CC-RL (V1.04.00)</td>
</tr>
<tr>
<td>EWRL78 (Ver. 2.21.1)</td>
</tr>
</tbody>
</table>

(2) Hardware development environment

<table>
<thead>
<tr>
<th>On-chip debug emulator</th>
<th>E1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcomputer used</td>
<td>RL78/F14(R5F10PLJ)</td>
</tr>
<tr>
<td>RL78/F14 mounted CPU board</td>
<td>- ECU001-F14-12V</td>
</tr>
<tr>
<td></td>
<td>- Replacing the capacitor (47pF to 0.1uF)</td>
</tr>
<tr>
<td></td>
<td>Target: C20, C22, and C24</td>
</tr>
<tr>
<td>BLDC motor</td>
<td>BLY171S-15V-8000</td>
</tr>
</tbody>
</table>
2. **System overview**
   
   Overview of this system is explained below.

2.1 **Hardware configuration**
   
   Hardware configuration is shown below.

![Figure 2-1 Hardware configuration Diagram](image-url)
2.2 Hardware specifications

2.2.1 Terminal interface

List of user interface of this system is given in Table 2-1.

<table>
<thead>
<tr>
<th>Terminal name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>P83 / ANI5</td>
<td>U phase voltage measurement</td>
</tr>
<tr>
<td>P84 / ANI6</td>
<td>V phase voltage measurement</td>
</tr>
<tr>
<td>P85 / ANI7</td>
<td>W phase voltage measurement</td>
</tr>
<tr>
<td>P86 / ANI8</td>
<td>VDC voltage measurement</td>
</tr>
<tr>
<td>P125 / TRDIOB0</td>
<td>Complementary PWM output (U_p)</td>
</tr>
<tr>
<td>P120 / TRDIOD0</td>
<td>Complementary PWM output (U_n)</td>
</tr>
<tr>
<td>P15 / TRDIOA1</td>
<td>Complementary PWM output (V_p)</td>
</tr>
<tr>
<td>P17 / TRDIOB1</td>
<td>Complementary PWM output (V_n)</td>
</tr>
<tr>
<td>P16 / TRDIOC1</td>
<td>Complementary PWM output (W_p)</td>
</tr>
<tr>
<td>P30 / TRDIOD1</td>
<td>Complementary PWM output (W_n)</td>
</tr>
<tr>
<td>P60</td>
<td>ERR1 input</td>
</tr>
<tr>
<td>P61</td>
<td>ERR2 input</td>
</tr>
<tr>
<td>P140</td>
<td>MUTE output</td>
</tr>
<tr>
<td>P31</td>
<td>SW Input</td>
</tr>
</tbody>
</table>

2.2.2 Peripheral functions

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

Please refer to “section 4 Description of peripheral functions” for details.

<table>
<thead>
<tr>
<th>Peripheral function</th>
<th>Usage</th>
</tr>
</thead>
</table>
| A/D converter (ANI5, ANI6, ANI7, ANI8) | - U, V and W phase voltage measurement  
- Bus voltage measurement |
| Timer RD (TRD)      | PWM output using complementary PWM mode (3 positive phases, 3 negative phases) |
| Port (P60, P61)     | error detection (Over current detection, low voltage detection, heating load short detection) |
| Port (P15, P16, P17, P30, P120, P125, P140) | - motor control signal with port output  
- MUTE terminal control signal output |
| Timer Array Unit (TAU) | - 1 [ms] interval timer  
- Free-run timer for speed measurement |
2.3 Software structure

2.3.1 Software file structure

Folders and files structure of the sample program is given below.

<table>
<thead>
<tr>
<th>inc</th>
<th>Table 2-3 Folders and files structure of sample program</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL78F14_120_SSLS_BEMF_InterMidi</td>
<td><strong>inc</strong></td>
</tr>
<tr>
<td></td>
<td><code>lodefine.h</code></td>
</tr>
<tr>
<td></td>
<td><code>main.h</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ctrl_board_interface.h</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ctrl_rl78f14.h</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ssns_less_120.h</code></td>
</tr>
<tr>
<td></td>
<td><code>RL78_mtr_common.h</code></td>
</tr>
<tr>
<td>src</td>
<td><strong>src</strong></td>
</tr>
<tr>
<td></td>
<td><code>main.c</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ctrl_board_interface.c</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ctrl_rl78f14.c</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_interrupt.c</code></td>
</tr>
<tr>
<td></td>
<td><code>mtr_ssns_less_120.c</code></td>
</tr>
<tr>
<td>asm</td>
<td><strong>asm</strong></td>
</tr>
<tr>
<td></td>
<td><code>cstat.asm</code></td>
</tr>
<tr>
<td></td>
<td><code>hwinit.asm</code></td>
</tr>
<tr>
<td></td>
<td><code>stkinit.asm</code></td>
</tr>
</tbody>
</table>
2.3.2 Modules structure

Module structure of the sample program is described below.

2.4 Software specifications

Basic specifications of software of this system are given in Table 2-4.

<table>
<thead>
<tr>
<th>Item</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control method</td>
<td>120-deg conducting method</td>
</tr>
<tr>
<td>Motor rotation start / stop</td>
<td>- Start by SW1 push down short time.</td>
</tr>
<tr>
<td></td>
<td>- Stop by SW1 push down long time or driver error detection.</td>
</tr>
<tr>
<td>Position detection of rotor magnetic pole</td>
<td>Position detection by back EMF (every 60 degrees)</td>
</tr>
<tr>
<td>Carrier frequency (PWM)</td>
<td>20 [kHz]</td>
</tr>
<tr>
<td>Control cycle</td>
<td>- Execute zero cross detection from back EMF per carrier frequency.</td>
</tr>
<tr>
<td></td>
<td>- Determination of PWM duty setting and conducting pattern.</td>
</tr>
<tr>
<td>Rotation speed control range</td>
<td>500 [rpm] to 5000 [rpm] both CW / CCW (8 Pole)</td>
</tr>
<tr>
<td>Rotation speed operation</td>
<td>- In pattern change, calculate rotate speed from elapse time of previous one.</td>
</tr>
<tr>
<td></td>
<td>- Uses the interval timer for measurement of elapse time.</td>
</tr>
<tr>
<td>Speed control (Speed PI control)</td>
<td>Obtains the speed command value form speed command value setting function, and performs speed control by PI control (5 [ms] cycle).</td>
</tr>
<tr>
<td>Processing stop for protection</td>
<td>Disables the motor control signal output (six outputs) under any of the following 3 conditions:</td>
</tr>
<tr>
<td></td>
<td>1. Rotation speed exceeds 33000[rpm] (electrical angle). (Monitored for each 1 [ms])</td>
</tr>
<tr>
<td></td>
<td>2. No zero-cross detected for 20 [ms] in sensorless drive mode.</td>
</tr>
<tr>
<td></td>
<td>3. Detect error signal (err1, err2) from pre-driver.</td>
</tr>
</tbody>
</table>
3. Motor control method

Sensorless 120 degrees conducting control and speed control of the BLDC motor, used in the sample program are explained here.

3.1 Sensorless 120 degrees conducting control of the BLDC motor

The sensorless control does not have a sensor for obtaining the permanent magnetic position, and hence the alternative to the sensor is required. The sensorless control of permanent magnetic synchronous motor, generally estimates the position by detecting the induced voltage (back-EMF).

The induced voltage in an closed circuit is proportion to the time rate of change of the magnetic flux through the circuit.

For example, consider the case where magnet gets close to the coil, as shown in Figure 3-1. In this case, since the interlinkage magnetic flux increase within the coil, coil generates the electromotive force that flows the current to prevent the increase of interlinkage magnetic flux in the direction of the figure. (The flux of opposite direction of the magnetic flux is occurred by the right-handed screw rule.)

![Figure 3-1 Induced voltage depending on the coil magnet](image)

This induced voltage $E_m$ is expressed by the magnetic flux $\phi_m$ as the following formula.

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

This event occurs event in the rotating permanent magnetic synchronous motor. When the permanent magnet is rotating, the induced voltage is generated by constantly changing interlinkage magnetic flux of each phase.
Figure 3-2 Induced voltage in the rotating permanent magnetic synchronous motor

Figure 3-3 shows the variation of interlinkage magnetic flux in the U phase. Size of the interlinkage magnetic flux is shown on the vertical (Y) axis and phase of the permanent magnet is shown on the horizontal (X) axis. Also position for disposing the N pole of permanent magnet to coil is considered as $\theta = 0$.

Figure 3-3 Variation of interlinkage magnetic flux

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

If considered similarly for V phase, V phase and W phase deviate respectively by $2\pi/3$, $4\pi/3$ phase from U phase. The interlinkage magnetic flux of the three phases is expressed by the following formula.
Also, the induced voltage of three phases is expressed by the following formula, by using formula (1), when the angle speed is considered as $\omega$.

\[
\begin{align*}
E_u &= \frac{d}{dt}\varphi_u = \frac{d}{dt}\varphi_m \cos \theta = -\omega \varphi_m \sin \theta = \omega \varphi_m \cos (\theta + \frac{\pi}{2}) \\
E_v &= \frac{d}{dt}\varphi_v = \frac{d}{dt}\varphi_m \cos (\theta - \frac{2}{3} \pi) = -\omega \varphi_m \sin (\theta - \frac{2}{3} \pi) = \omega \varphi_m \cos (\theta - \frac{\pi}{6}) \\
E_w &= \frac{d}{dt}\varphi_w = \frac{d}{dt}\varphi_m \cos (\theta - \frac{4}{3} \pi) = -\omega \varphi_m \sin (\theta - \frac{4}{3} \pi) = \omega \varphi_m \cos (\theta - \frac{5}{6} \pi)
\end{align*}
\]

From this formula, it is understood that the induced voltage lead of $\pi / 2$ phase from permanent magnetic flux. This mean that if the induced voltage can be detected, position the permanent magnet can be estimated.

**Figure 3-4 Zero-crossing of the induced voltage**

However, the induced voltage of each phase is not always detected while the motor is rotating. During the driving in 120 degrees conduction, conduction is performed to the two phases among the three phases and hence only the remaining one phase, to which conduction is not performed, can detect the induced voltage. Actually, position information is obtained by detecting the point of change in the sign of induced voltage (zero-crossing) occurring in mon-conducting phase, which can detect the induced voltage.
In the three phases motor, this zero-crossing occurs for total six times, i.e. twice in each phase, in one rotation (electrical angle) of the motor. This means that the position for every 60 degrees can be detected by this process in the same way as resolution of hall sensor.

Figure 3-5 Relation between conducting pattern and zero-crossing (Upper arm chopping)

However, this zero-crossing detection signal cannot be used in the same way as the signal of the hall sensor.

The zero-crossing detection signal occurs at the point where phase is shifted $\frac{\pi}{6}$ from proper conducting pattern switching timing, as shown in Figure 3-5. Therefore, in the actual control, conducting pattern is switched at the point where phase is shifted $\frac{\pi}{6}$ from detecting the zero-crossing.
3.2 Zero-crossing detection method

Various zero-crossing detection methods are used. The method of detecting the zero-crossing by comparing the value of induced voltage with the center point voltage by the software, using the A/D converter of microcomputer is introduced here. Since voltage is compared without the comparator, it is called as comparator less method.

![Figure 3-6 Comparator less method](image)

Actually detecting the induced voltage, commutation voltage occurring when switching the conducting patterns, and impact of the PWM of other phases must be considered. This impact is expressed in the format shown in Figure 3-7.

![Figure 3-7 Overview diagram of impact of the commutation and other phase PWM](image)

In this system, impact is removed by using the simple filter route and the software.
3.3 Start-up method

Induced voltage dues not occur unless the permanent magnet is rotating. This means that the position of magnet cannot be estimated by using the induced voltage, at the time of starting.

Therefore, start-up method in this system synchronizes speed of the permanent magnet by generating a routing magnetic field by forcibly switching the conducting pattern regardless of the position of permanent magnet.

Figure 3-8 Diagram of start-up operation
3.4 Position estimate operation

In this system, the virtual center voltage of the motor is calculated by the sum of A/D conversion voltage of each phase in each PWM control cycle. The pattern of ‘1’ ‘0’ is created by comparing the virtual motor center voltage with each phase voltage.

Then, the conduction pattern is created by shifted $\pi/6$.

$\pi/6$ is estimated from the number of PWM control between zero-crossing.

![Figure 3-9 conduction pattern (Upper arm chopping)](image)
3.5 Speed control

In this system, the motor rotation speed is calculated from a difference between the previous and current timer values by detecting the zero-crossing, at the time of switching the patterns while having the timer of channel 1 of timer array unit performed free running.

Furthermore, in this system, the calculation result is processed LPF.

This system is using PI control for speed control. A voltage command value at any (discrete) time ‘n’ is calculated by the following formula.

\[
V[n] = V[n-1] + K_p \times (err[n] - err[n-1]) + K_i \times err[n]
\]

**V**: Command value voltage  \( \text{err} \): Deviation of rotation speed command value and rotation speed calculation value  \( K_p \): Proportional gain  \( K_i \): Integral gain
Also, PWM control is used for the output voltage control. The PWM control is a control method that continuously adjusts the average voltage by varying the duty of pulse, as shown Figure 3-11.

![Figure 3-11 PWM control](image)

Modulation factor ‘m’ is defined as follows.

\[
m = \frac{V}{E}
\]

- \(m\): Modulation factor
- \(V\): Command value voltage
- \(E\): Inverter bus voltage

This modulation factor is reflected in the setting value of resistor that determines the PWM duty.

In this system, complementary PWM chopping (120 degrees) is adopted and thus output voltage and speed are controlled. An example of motor control signal output waveforms at the time of complementary PWM is given in Figure 3-12.

![Figure 3-12 Waveform of complementary PWM chopping (120 degrees)](image)
4. Description of peripheral functions used

Peripheral functions used in this system are explained. Following peripheral functions are explained in this chapter.

- External interruption function
- A/D converter
- Timer Array Unit TAUS function
- Timer RD function

4.1 External interruption function

In this system, external interruptions are set as given in Table 4-1.

<table>
<thead>
<tr>
<th>Interruption</th>
<th>Item</th>
<th>Content</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTP0</td>
<td>Valid edge</td>
<td>Falling edge</td>
<td>Over current detection</td>
</tr>
<tr>
<td></td>
<td>Interruption priority level</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

4.2 A/D converter function

A/D converter converts the analog input to digital value. The target microcontroller (RL78/F14), incorporates one circuit of 10bit A/D converter. Analog input of twelve channels can be converted to digital values by controlling the conversion channel.

In this system, the A/D converter is set as given in Table 4-2.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Item</th>
<th>Content</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANI5, ANI6, ANI7</td>
<td>Conversion time</td>
<td>3.563 [us]</td>
<td>U, V, W phase voltage</td>
</tr>
<tr>
<td></td>
<td>Channel selection mode</td>
<td>Select mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion operation mode</td>
<td>One-shot conversion mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion starting conditions</td>
<td>Software trigger</td>
<td></td>
</tr>
<tr>
<td>ANI8</td>
<td>Conversion time</td>
<td>3.563 [us]</td>
<td>Inverter bus voltage</td>
</tr>
<tr>
<td></td>
<td>Channel selection mode</td>
<td>Select mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion operation mode</td>
<td>One-shot conversion mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conversion starting conditions</td>
<td>Software trigger</td>
<td></td>
</tr>
</tbody>
</table>
4.3 Timer Array Unit TAUS function

The Timer Array Unit TAUS consists of four 16bit timers. Each 16-bit timer called ‘Channel’ and can be used as an independent timer as well as an advanced timer function by combining multiple channels.

![Figure 4-1 Timer Array Unit](image)

In this system, the Timer Array Unit is set as given in Table 4-3.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Item</th>
<th>Content</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 0</td>
<td>Operation mode of timer</td>
<td>Interval timer</td>
<td>Timer for generating 1[ms]</td>
</tr>
<tr>
<td></td>
<td>Source clock</td>
<td>CK00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Count clock frequency</td>
<td>24 [MHz]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Interruption cycle</td>
<td>1 [ms]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Setting value of Timer data resistor 0 (TDR00)</td>
<td>23999 (1[ms]/(1/24[MHz])) - 1)</td>
<td></td>
</tr>
<tr>
<td>Channel 1</td>
<td>Operation mode of timer</td>
<td>Interval timer</td>
<td>Timer for speed calculation</td>
</tr>
<tr>
<td></td>
<td>Source clock</td>
<td>CK01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Count clock frequency</td>
<td>125 [kHz]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Interruption cycle</td>
<td>524 [ms] (unused)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Setting value of Timer data resistor 0 (TDR00)</td>
<td>65535</td>
<td></td>
</tr>
</tbody>
</table>
Also, basic timings of the interval timer are shown in Figure 4-2.

Figure 4-2 Example of basic timings of interval timer (Example of Channel 0)
4.4 Timer RD function

Timer RD has two 16-bit timers (timer RD0 and timer RD1).

Following four modes are provided in timer RD.

- Timer mode
- Reset synchronous PWM mode
- Complementary PWM mode
- PWM3 mode

In this system, the timer RD is set as given in Table 4-4.

Table 4-4 Timer RD setting details

<table>
<thead>
<tr>
<th>Timer used</th>
<th>Item</th>
<th>Content</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer RD</td>
<td>Mode used</td>
<td>Complementary PWM mode</td>
<td>6 phase PWM output</td>
</tr>
<tr>
<td></td>
<td>PWM cycle</td>
<td>50 [us]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dead time</td>
<td>2.0 [us]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Count frequency</td>
<td>48 [MHz]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output level</td>
<td>Initial output is &quot;Low&quot;, Active level is &quot;High&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Buffer operation</td>
<td>Valid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pulse output forced</td>
<td>Valid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>shutdown control</td>
<td>(Output value at the time of shutdown: Hi-Z)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output port</td>
<td>Refer to Figure 4-3</td>
<td></td>
</tr>
</tbody>
</table>

Note:

In complementary PWM mode, the timer RD outputs a waveform by combining the counters and resistors of timer RD0 and timer RD1.
An example of PWM output waveform is shown in Figure 4-3.

Figure 4-3 Example of PWM output waveform in complementary PWM mode

4.5 Calculation of PWM duty setting using modulation factor

This part summarizes how to set duty in complementary PWM mode.

As first, calculate positive phase active level width by using modulation rate in section 3.5. Next, calculate setting value of TRDGRB0, TRDGRA1, and TRDGRB1 registers that output positive phase active level width.

Positive phase active level width = PWM cycle * modulation rate
TRDGRB0 = TRDGRA1 = TRDGRB1 = TRDGRA1 - TRD0 + 1 - Positive phase active level width
5. Description of control program

Control program of this system is explained here.

5.1 Control block diagram

In the sample program, a motor is driven by open loop control. After that, control is performed according to the following block diagram.

![Control block diagram](image)

<table>
<thead>
<tr>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Rotation speed</td>
</tr>
<tr>
<td>Nrkt</td>
<td>Rotation speed after LPF</td>
</tr>
<tr>
<td>N*</td>
<td>Rotation speed command value</td>
</tr>
<tr>
<td>Nerr</td>
<td>Rotation speed deviation</td>
</tr>
<tr>
<td>V*</td>
<td>Voltage command value</td>
</tr>
<tr>
<td>Vdc</td>
<td>Inverter bus voltage</td>
</tr>
<tr>
<td>m</td>
<td>Modulation factor</td>
</tr>
<tr>
<td>PWM</td>
<td>PWM output signal</td>
</tr>
<tr>
<td>Vu, Vv, Vw</td>
<td>Phase voltage</td>
</tr>
<tr>
<td>HU, HV, HW</td>
<td>Hall sensor signal</td>
</tr>
<tr>
<td>θ</td>
<td>Rotor position</td>
</tr>
</tbody>
</table>
Function is given below.

(1) Position detection of permanent magnet
   Permanent magnetic position is detected by the zero-crossing of each phase of U, V and W.

(2) Rotation speed calculation
   Rotation speed is calculated from the timer counter (TCR01) at the timing of detecting zero-crossing. The rotation speed calculation value is used in speed control.

(3) Speed control
   Speed control is using PI control. The output value of speed PI control is set as a voltage command value.

(4) Processing stop for protection
   Processing stop for protection prevents the motor of inverter from breakage due to over current, over voltage, and over speed.
5.2 Contents of control

5.2.1 Motor start/stop

Starting and stopping the motor are controlled by input from SW1.

A general-purpose port (P31) is assigned to SW1. The sample program reads the P31 port within the main loop. When SW1 is pushed short time, it is judged that the start switch is on. On the other hand, when SW1 is pushed long time, the program determined to stop the motor.

When push SW1 while the motor is start, a rotation speed command value is changed by software.

![Figure 5-2 Conceptual diagram of start switch external circuit](image)

5.2.2 Inverter bus voltage

Inverter bus voltage is measured as given in Table 5-1. It is use for modulation factor calculation san overvoltage detection. (When an overvoltage is detected, PWM is stopped.)

<table>
<thead>
<tr>
<th>Item</th>
<th>Conversion ratio (Inverter voltage Vdc: A/D conversion value)</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverter bus voltage</td>
<td>0 [V] to 26 [V] : 000H to 03FFH</td>
<td>AN18</td>
</tr>
</tbody>
</table>

![Figure 5-3 Conceptual diagram of inverter voltage measurement external circuit](image)
5.2.3 3 phase voltage of motor

(1) U, V, W phase voltage
Voltage of U, V and W phase is calculated and is used for zero-crossing judgment as shown in Table 5-2.

<table>
<thead>
<tr>
<th>Item</th>
<th>Conversion ratio (Inverter voltage Vdc: A/D conversion value)</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>U, V, W phase voltage</td>
<td>0 [V] to 15 [V] : 0000H~03FFH</td>
<td>ANI5, ANI6, ANI7</td>
</tr>
</tbody>
</table>

Figure 5-4 Conceptual diagram of U, V and W phase voltage measurement external circuit
5.2.4 Rotation speed operations

The rotation speed is calculated by using zero-crossing detection and free run timer (TAUS channel 01). In the timing of pattern switching by zero-crossing detection, counter value of free run timer is obtained from on the difference with the previously obtained value. Based on the difference speed is calculated by the following formula.

Rotation speed (N) = (60 * 125 [kHz]) / \{(last counter value – current counter value) * 6\}

Notes:
1. 125 [kHz] = (count click frequency of free-run timer)
2. (*6) is done at the period of obtaining the counter value is ‘\pi /3’.

In this sample program, LPF (migration average) processing is performed for the speed calculation result before speed PI control.

5.2.5 Speed PI control

In this sample program, speed PI control is performed on a 5 [ms] cycle, to avoid the multiple executions of PI control during hall sensor interruption. The voltage command value (V*) is created as given below.

Proportional (P) term: KP * (current rotation speed deviation – last rotation speed deviation)
Integral (I) term: KI * (current rotation speed deviation)

Voltage command value (V*) = previous voltage command value + proportional term + integral term

Notes:
1. Proportional gain (KP): 0.0001
2. Integral gain (KI): 0.000001
   Values of KP and KI depend on the used system.

For details of PI control, refer to specialized books.
5.2.6 System protection function

This control program has the following 5 types of error status and enables emergency stop functions in case of occurrence of respective error.

- Pre-driver error
  The error signal (ERR1 and ERR2) of over voltage error, under voltage error and short detection error from pre-driver is monitored with general ports, CPU performs emergency stop.
  The ERR1 use P60, the ERR2 use P61. A kind of external notification errors and combination of terminals are shown in Table 5-3.

<table>
<thead>
<tr>
<th>A kind of errors</th>
<th>ERR1 (P60)</th>
<th>ERR2 (P61)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Over voltage detection</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Under voltage detection</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>short detection</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>No Error</td>
<td>High</td>
<td>High</td>
</tr>
</tbody>
</table>

- Over current error
  An emergency stop signal (overcurrent detection) from hardware forces the program to execute high impedance output to PWM the output port (emergency stop without involving CPU). The INTP0 port is used.

- Rotation speed abnormality error
  The rotation speed calculation value is monitored with 1 [ms] interval. When an error value is detected in rotation speed values (in a case of value over 33000 [rpm] (electrical angle)), CPU performs emergency stop.

- Timeout error
  When no hall sensor interruption occurs for a certain period (20 [ms]), CPU performs emergency stop.

- Induced voltage signal pattern error
  The patterns created from each phase voltage of U, V and W are monitored. When an error pattern is detected, CPU performs emergency stop.
5.3 System resources

5.3.1 Interruption
List of interruptions used in this control program is given here.

<table>
<thead>
<tr>
<th>Interruption</th>
<th>Interruption handler</th>
<th>Interruption occurrence condition</th>
<th>Main function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrier synchronous (INTTRD0)</td>
<td>void mtr_carrier_interrupt(void)</td>
<td>50 [us] (20 [kHz])</td>
<td>- Zero-crossing detection processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Conducting pattern switching</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Rotation speed operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Clearing the motor stop determination counter value</td>
</tr>
<tr>
<td>Interval timer interruption</td>
<td>void mtr_tau0_interrupt(void)</td>
<td>1 [ms] (1 [kHz])</td>
<td>- Speed PI control</td>
</tr>
<tr>
<td>(INTTM00)</td>
<td></td>
<td></td>
<td>- Error monitoring</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Control start time measurement</td>
</tr>
<tr>
<td>Over current detection</td>
<td>void mtr_over_current_interrupt(void)</td>
<td>Over current detection</td>
<td>- Over current protection</td>
</tr>
<tr>
<td>interruption (INTP0)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.3.2 A/D converter input signal and used channels
List of used channels of A/D converter used in this control program is given below.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Measurement signal</th>
<th>Range of setting value</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANI5, ANI6, ANI7</td>
<td>Voltage of U, V, W phase</td>
<td>15 [V] / 5 [V]</td>
<td>Used in zero-crossing detection</td>
</tr>
<tr>
<td>ANI8</td>
<td>Inverter bus voltage</td>
<td>26 [V] / 5 [V]</td>
<td>Used in modulation factor calculation, over voltage protection</td>
</tr>
</tbody>
</table>
5.3.3 Port function

List of port functions used in this control program is given below.

<table>
<thead>
<tr>
<th>Input / output</th>
<th>Port number</th>
<th>Function</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>P60</td>
<td>ERR1 port input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P61</td>
<td>ERR2 port input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P31</td>
<td>SW1 port input</td>
<td>Change motor speed</td>
</tr>
<tr>
<td>Output</td>
<td>P140</td>
<td>MUTE port control signal output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P125</td>
<td>U phase upper arm motor control signal port output</td>
<td>(Up) Logic setting is ‘High’ active.</td>
</tr>
<tr>
<td></td>
<td>P120</td>
<td>U phase lower arm motor control signal port output</td>
<td>(Un)</td>
</tr>
<tr>
<td></td>
<td>P15</td>
<td>V phase upper arm motor control signal port output</td>
<td>(Vp)</td>
</tr>
<tr>
<td></td>
<td>P16</td>
<td>V phase lower arm motor control signal port output</td>
<td>(Vn)</td>
</tr>
<tr>
<td></td>
<td>P17</td>
<td>W phase upper arm motor control signal port output</td>
<td>(Wp)</td>
</tr>
<tr>
<td></td>
<td>P30</td>
<td>W phase lower arm motor control signal port output</td>
<td>(Wn)</td>
</tr>
</tbody>
</table>

5.3.4 PWM output part

List of PWM output used in this control program is below.

<table>
<thead>
<tr>
<th>Input / output</th>
<th>Output port</th>
<th>Function</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>TRDIOB0</td>
<td>U phase upper arm motor control signal PWM output</td>
<td>(Up) Logical setting is ‘High’ active.</td>
</tr>
<tr>
<td></td>
<td>TRDIOA0</td>
<td>U phase lower arm motor control signal PWM output</td>
<td>(Un)</td>
</tr>
<tr>
<td></td>
<td>TRDIOA1</td>
<td>V phase upper arm motor control signal PWM output</td>
<td>(Vp)</td>
</tr>
<tr>
<td></td>
<td>TRDIOC1</td>
<td>V phase lower arm motor control signal PWM output</td>
<td>(Vn)</td>
</tr>
<tr>
<td></td>
<td>TRDIOB1</td>
<td>W phase upper arm motor control signal PWM output</td>
<td>(Wp)</td>
</tr>
<tr>
<td></td>
<td>TRDIOA1</td>
<td>W phase lower arm motor control signal PWM output</td>
<td>(Wn)</td>
</tr>
</tbody>
</table>
5.4 Function specifications

Multiple control functions are used in this control program. Lists of control functions are given below.
For detailed processing, refer to flowcharts or source files.

<table>
<thead>
<tr>
<th>File name</th>
<th>Function overview</th>
<th>Processing overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>main.c</td>
<td>main()</td>
<td>- Hardware initialization function call</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td>- User interface initialization function call</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td>- Main processing used variable initialization function call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- status transition and event execution function call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Main processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-&gt; Main processing execution function call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-&gt; Watchdog timer clear function call</td>
</tr>
<tr>
<td>ctrl_ui()</td>
<td>Input: None</td>
<td>- Motor status change</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td>- Determination of rotation speed command value and rotation direction</td>
</tr>
<tr>
<td>software_init()</td>
<td>Input: None</td>
<td>Initialization of variables used in the main processing</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td>check_sw()</td>
<td>Input: sw_mode / current SW mode</td>
<td>Obtaining the SW pushed status.</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td>change_ref_speed_in_stages()</td>
<td>Input: None</td>
<td>Change reference speed in stages.</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td>mtr_ctrl_board_interface.c</td>
<td>get_sw1()</td>
<td>Obtaining the status of SW1.</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: (uint8) tmp_port / level of SW1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_ssns_less_120.c</td>
<td>R_MTR_InitSequence()</td>
<td>Initialization for variables to use for sequence control</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R_MTR_ExecEvent()</td>
<td>- Changing the status</td>
</tr>
<tr>
<td></td>
<td>Input: (uint8)u1_event / Occurred event</td>
<td>- Calling an appropriate processing execution function for the occurred event</td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td>mtr_act_run()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>- Variable initialization function call upon motor startup</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td>- Motor control startup function call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Output pattern determination function call</td>
</tr>
<tr>
<td>mtr_act_stop()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>Motor control stop function call</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td></td>
</tr>
<tr>
<td>mtr_act_none()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>No processing is performed</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td></td>
</tr>
<tr>
<td>mtr_act_reset()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>- Global variable initialization</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td>- Wait motor stop</td>
</tr>
<tr>
<td>mtr_act_error()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>Motor control stop function call</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td></td>
</tr>
<tr>
<td>mtr_pattern_set()</td>
<td>Input: (uint8)u1_state / motor status</td>
<td>- Speed measurement function call</td>
</tr>
<tr>
<td></td>
<td>Output: (uint8)u1_state / motor status</td>
<td>- Conducting pattern determination</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Motor control signal creation function call</td>
</tr>
</tbody>
</table>
## Table 5-8 List of control functions (2/3)

<table>
<thead>
<tr>
<th>File name</th>
<th>Function overview</th>
<th>Processing overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtr_speed_calc()</td>
<td>Speed measurement calculation processing</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_start_init()</td>
<td>Initializing only the variables required for motor startup</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_pi_ctrl_speed()</td>
<td>Speed PI control</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R_MTR_SetSpeed()</td>
<td>Rotation speed command value setting</td>
<td></td>
</tr>
<tr>
<td>Input: (int16)ref_speed / Rotation speed command value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R_MTR_SetDir()</td>
<td>Rotation direction setting</td>
<td></td>
</tr>
<tr>
<td>Input: (uint8)dir / Rotation direction command value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R_MTR_GetSpeed()</td>
<td>Obtaining the rotation speed calculation value</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: (int16)g_s2_rpm / Rotation speed calculation value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R_MTR_GetStatus()</td>
<td>Obtaining the motor status</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: (uint8)g_u1_mode_system / Motor status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_error_check()</td>
<td>Error monitoring and detection</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_detect_zerocross()</td>
<td>- zero-crossing detection -create timing of conducting pattern modification</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_25108_err()</td>
<td>Error detection from pre-driver (overvoltage, under voltage, short detection)</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_over_current_interrupt ()</td>
<td>- Motor control error function call</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_tau0_interrupt()</td>
<td>- Error check Function call - Calling speed PI control function every 5 [ms] - Open loop starting control</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtr_carrier_interrupt()</td>
<td>- zero-crossing detection function call - Compare match flag (IFMA) clear function call</td>
<td></td>
</tr>
<tr>
<td>Input: None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output: None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-8 List of control functions (3/3)

<table>
<thead>
<tr>
<th>File name</th>
<th>Function overview</th>
<th>Processing overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtr_ctrl_rlf14.c</td>
<td><strong>R_MTR_InitHardware()</strong></td>
<td>Initializing clock and peripheral functions</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitClock()</strong></td>
<td>Initializing clock</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitPort()</strong></td>
<td>Initializing I/O ports</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitTAU()</strong></td>
<td>Initializing timer array unit</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitTRD()</strong></td>
<td>Initializing timer RD</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitADC()</strong></td>
<td>Initializing A/D convertor</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>R_MTR_InitExtInt()</strong></td>
<td>Initializing external interrupts</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>init_ui()</strong></td>
<td>Initializing user usage peripheral functions</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>mtr_ctrl_start()</strong></td>
<td>Starting TAU0</td>
</tr>
<tr>
<td></td>
<td>Input: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output: None</td>
<td></td>
</tr>
</tbody>
</table>
|                   | **mtr_ctrl_stop()**                                          | - Stopping timer RD                     
|                   | Input: None                                                  | - Stopping TAU0                                                                       |
|                   | Output: None                                                 | - Changing the motor control output port to inactive status                         |
|                   | **mtr_ctrl_error()**                                         | - Waiting motor stop                                                                  |
|                   | Input: None                                                  |                                                                                     |
|                   | Output: None                                                 |                                                                                     |
|                   | **mtr_change_pattern()**                                     | - Setting output pattern                                                            |
|                   | Input: (uint8)pattern / conducting pattern                   | - Changing the motor status when output pattern error occurs                         |
|                   | Output: None                                                 | - Event processing selection function call                                          |
|                   | **mtr_get_adc()**                                            | Executing A/D conversion                                                            |
|                   | Input: (uint8)ad_ch / Conversion channel                     |                                                                                     |
|                   | Output: (int16)s2_temp / A/D conversion result                |                                                                                     |
|                   | **clear_wdt()**                                               | Clearing the watchdog timer                                                         |
|                   | Input: None                                                  |                                                                                     |
|                   | Output: None                                                 |                                                                                     |
|                   | **mtr_clear_trd0_imfa()**                                     | Clearing the compare match flag (IMFA)                                              |
|                   | Input: None                                                  |                                                                                     |
|                   | Output: None                                                 |                                                                                     |
5.5 Variables list

Lists of variables used in this control program are given below. Note that local variables are not described.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Type</th>
<th>Content</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>g_u1_cnt_speed_pi</td>
<td>uint8</td>
<td>Speed PI control decimation counter</td>
<td>- Speed PI control cycle 5 [ms] is counted.</td>
</tr>
<tr>
<td>g_s2_pwm_duty</td>
<td>int16</td>
<td>Timer RD compare register setting value</td>
<td>-</td>
</tr>
<tr>
<td>g_u1_openloop_period</td>
<td>uint8</td>
<td>Period of openloop drive</td>
<td>-</td>
</tr>
<tr>
<td>g_u1_cnt_openloop_period</td>
<td>uint8</td>
<td>Counter of period for openloop drive</td>
<td>- Counter of 1[ms] interrupt</td>
</tr>
<tr>
<td>g_u1_cnt_openloop_pattern</td>
<td>uint8</td>
<td>Counter of changing pattern for openloop drive</td>
<td>-</td>
</tr>
<tr>
<td>g_u2_cnt_wait_stop</td>
<td>uint16</td>
<td>Counter for waiting for motor stop</td>
<td>Counter for waiting for motor stop</td>
</tr>
<tr>
<td>g_u1_flg_wait_stop</td>
<td>uint8</td>
<td>Motor stop waiting flag</td>
<td>0: Not motor stop waiting state</td>
</tr>
<tr>
<td>g_u2_run_mode</td>
<td>uint16</td>
<td>Operation mode management</td>
<td>1: Motor stop waiting state</td>
</tr>
<tr>
<td>g_u1_error_status</td>
<td>uint8</td>
<td>Error status management</td>
<td>0: Overcurrent error</td>
</tr>
<tr>
<td>g_u1_mode_system</td>
<td>uint8</td>
<td>State management</td>
<td>1: Overcurrent error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2: Overvoltage error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3: Over speed error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4: Timeout error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7: Induction voltage pattern error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8: Under voltage error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9: Short error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(0xff: Non-definition error)</td>
</tr>
</tbody>
</table>
### 5.6 Macro definitions

Lists of macro definitions used in this control program are given below.

Table 5-10 Macro definitions list (1/5)

<table>
<thead>
<tr>
<th>File name</th>
<th>Macro name</th>
<th>Definition value</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>main.h</td>
<td>SW_ON</td>
<td>0</td>
<td>active level</td>
</tr>
<tr>
<td></td>
<td>SW_OFF</td>
<td>1</td>
<td>inactive level</td>
</tr>
<tr>
<td></td>
<td>SW_MODE_NONE</td>
<td>0</td>
<td>SW is not pushed</td>
</tr>
<tr>
<td></td>
<td>SW_MODE_SHORT</td>
<td>1</td>
<td>SW pushed short time</td>
</tr>
<tr>
<td></td>
<td>SW_MODE_LONG</td>
<td>2</td>
<td>SW pushed long time</td>
</tr>
<tr>
<td></td>
<td>CHATTERING_CNT</td>
<td>100</td>
<td>Chattering removal</td>
</tr>
<tr>
<td></td>
<td>PUSH_CNT</td>
<td>0xF000</td>
<td>Using for judge SW1 pushed long time</td>
</tr>
<tr>
<td></td>
<td>PUSH_CNT_CLR</td>
<td>0x0000</td>
<td>Counter clear value</td>
</tr>
<tr>
<td></td>
<td>SOFT_STOP_SPEED</td>
<td>0</td>
<td>0 speed [rpm] (mechanical angle)</td>
</tr>
<tr>
<td></td>
<td>SOFT_MIN_SPEED</td>
<td>0</td>
<td>min speed [rpm] (mechanical angle)</td>
</tr>
<tr>
<td></td>
<td>SOFT_MAX_SPEED</td>
<td>3000</td>
<td>max speed [rpm] (mechanical angle)</td>
</tr>
<tr>
<td></td>
<td>SOFT_DIFF_SPEED</td>
<td>500</td>
<td>increment / decrement difference speed [rpm] (mechanical angle)</td>
</tr>
<tr>
<td></td>
<td>REF_SPEED_DECEL</td>
<td>0</td>
<td>deceleration reference speed</td>
</tr>
<tr>
<td></td>
<td>REF_SPEED_ACCEL</td>
<td>1</td>
<td>acceleration reference speed</td>
</tr>
<tr>
<td>File name</td>
<td>Macro name</td>
<td>Definition value</td>
<td>Remark</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------</td>
<td>------------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td>mtr_ctrl_rl78f14.h</td>
<td>MTR_PWM_TIMER_FREQ</td>
<td>48</td>
<td>Timer RD count frequency [MHz]</td>
</tr>
<tr>
<td></td>
<td>MTR_TAU1_FREQ</td>
<td>93750</td>
<td>Timer Array Unit channel 1 count frequency [Hz]</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_UP</td>
<td>P12.5</td>
<td>U phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_UN</td>
<td>P12.0</td>
<td>U phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_VP</td>
<td>P1.5</td>
<td>V phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_VN</td>
<td>P1.6</td>
<td>V phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_WP</td>
<td>P1.7</td>
<td>W phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_WN</td>
<td>P3.0</td>
<td>W phase mode output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_CTL_UP</td>
<td>PMC12.0</td>
<td>PWM mode control</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_CTL_UN</td>
<td>PMC12.5</td>
<td>PWM mode control</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_UP</td>
<td>P12.5</td>
<td>U phase (positive phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_UN</td>
<td>P12.0</td>
<td>U phase (negative phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_VP</td>
<td>P1.5</td>
<td>V phase (positive phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_VN</td>
<td>P1.6</td>
<td>V phase (negative phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_WP</td>
<td>P1.7</td>
<td>W phase (positive phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_WN</td>
<td>P3.0</td>
<td>W phase (negative phase) output port</td>
</tr>
<tr>
<td></td>
<td>MTR_TAU1_CNT</td>
<td>TCR01</td>
<td>Timer count register for speed measurement</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ERR1</td>
<td>PM6.0</td>
<td>input port mode of ERR1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ERR2</td>
<td>PM6.1</td>
<td>input port mode of ERR2</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_ERR1</td>
<td>P6.0</td>
<td>input port of ERR1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_ERR2</td>
<td>P6.1</td>
<td>input port of ERR2</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_PULLUP_ERR1</td>
<td>PU6.0</td>
<td>port pull up of ERR1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_PULLUP_ERR2</td>
<td>PU6.1</td>
<td>port pull up of ERR2</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_SW1</td>
<td>PM3.1</td>
<td>input port mode of SW1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_SW1</td>
<td>P3.1</td>
<td>input port of SW1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_PULLUP_SW1</td>
<td>PU3.1</td>
<td>input port pullup of SW1</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_MUTE</td>
<td>PM14.0</td>
<td>output port mode of MUTE</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MUTE</td>
<td>P14.0</td>
<td>output port of MUTE</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ADCCH_VU</td>
<td>PM8.3</td>
<td>input port mode of ADC (VU)</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ADCCH_VV</td>
<td>PM8.4</td>
<td>input port mode of ADC (VV)</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ADCCH_VW</td>
<td>PM8.5</td>
<td>input port mode of ADC (VW)</td>
</tr>
<tr>
<td></td>
<td>MTR_PORT_MODE_ADCCH_VDC</td>
<td>PM8.6</td>
<td>input port mode of ADC (VDC)</td>
</tr>
<tr>
<td></td>
<td>MTR_ADCCH_VDC</td>
<td>8</td>
<td>VDC voltage A/D conversion channel</td>
</tr>
<tr>
<td></td>
<td>MTR_ADCCH_VU</td>
<td>5</td>
<td>U phase voltage A/D conversion channel</td>
</tr>
<tr>
<td></td>
<td>MTR_ADCCH_VV</td>
<td>6</td>
<td>V phase voltage A/D conversion channel</td>
</tr>
<tr>
<td></td>
<td>MTR_ADCCH_VW</td>
<td>7</td>
<td>W phase voltage A/D conversion channel</td>
</tr>
<tr>
<td></td>
<td>MTR_MAX_VDC</td>
<td>12</td>
<td>Limit of the voltage command value [V]</td>
</tr>
<tr>
<td></td>
<td>MTR_VDC_RESOLUTION</td>
<td>26 / 1023</td>
<td>Inverter bus voltage resolution</td>
</tr>
<tr>
<td>File name</td>
<td>Macro name</td>
<td>Definition value</td>
<td>Remark</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------</td>
<td>------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>mtr_sns_sens_120.h</td>
<td>MTR_CARRIER_FREQ</td>
<td>20.0f</td>
<td>PWM carrier frequency [kHz]</td>
</tr>
<tr>
<td></td>
<td>MTR_DEADTIME_US</td>
<td>2.0f</td>
<td>Dead time value [us]</td>
</tr>
<tr>
<td></td>
<td>MTR_START_DUTY</td>
<td>8.0f</td>
<td>PWM duty initial value [%]</td>
</tr>
<tr>
<td></td>
<td>MTR_OPENLOOP_START_PERIOD</td>
<td>3</td>
<td>Default open-loop drive period</td>
</tr>
<tr>
<td></td>
<td>MTR_OPENLOOP_CHANGE_CNT</td>
<td>84</td>
<td>The number of pattern changes in the same frequency (open-loop)</td>
</tr>
<tr>
<td></td>
<td>MTR_CHANGE_MODE_PERIOD</td>
<td>2</td>
<td>changing period open-loop drive to sensorless drive</td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_U_V</td>
<td>3</td>
<td>CW conducting pattern value</td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_U_W</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_V_W</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_V_U</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_W_U</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CW_W_V</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_U_V</td>
<td>2</td>
<td>CCW conducting pattern value</td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_W_V</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_W_U</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_V_U</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_V_W</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_PATTERN_CCW_U_W</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-10 Macro definitions list (4/5)

<table>
<thead>
<tr>
<th>File name</th>
<th>Macro name</th>
<th>Definition value</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtr_ssns_less_120.h</td>
<td>MTR_SPEED_PI_DECIMATION</td>
<td>4</td>
<td>Speed PI control decimation count</td>
</tr>
<tr>
<td></td>
<td>MTR_SPEED_PI_KP</td>
<td>0.0001f</td>
<td>Proportional term gain</td>
</tr>
<tr>
<td></td>
<td>MTR_SPEED_PI_KI</td>
<td>0.00001f</td>
<td>Integral term gain</td>
</tr>
<tr>
<td></td>
<td>MTR_AVG_OLD</td>
<td>0.3f</td>
<td>LPF previous value filter coefficient</td>
</tr>
<tr>
<td></td>
<td>MTR_CARRIER_SET</td>
<td>(\frac{1000}{MTR_CARRIER_FREQ \times MTR_PWM_TIMER_FREQ - 1})</td>
<td>Set value of carrier wave frequency</td>
</tr>
<tr>
<td></td>
<td>MTR_PWM_DEAD_TIME</td>
<td>(MTR_PWM_TIMER_FREQ \times MTR_DEADTIME_US)</td>
<td>Dead time (period)</td>
</tr>
<tr>
<td></td>
<td>MTR_START_DUTY_SET</td>
<td>(\frac{(((MTR_CARRIER_SET + 1) / 100) \times MTR_START_DUTY)}{2} + MTR_PWM_DEAD_TIME - 1)</td>
<td>Set value of default duty</td>
</tr>
<tr>
<td></td>
<td>MTR_PWM_PERIOD</td>
<td>(\frac{(((MTR_CARRIER_SET + 1) / 2) + MTR_PWM_DEAD_TIME)}{100})</td>
<td>Set value of PWM period</td>
</tr>
<tr>
<td></td>
<td>MTR_RATE_DUTY</td>
<td>(MTR_START_DUTY / 100)</td>
<td>Set value of initial voltage</td>
</tr>
<tr>
<td></td>
<td>MTR_MAX_PWM_DUTY</td>
<td>(\frac{(((MTR_CARRIER_SET + 1) / 100) \times 95)}{100} - 1)</td>
<td>Max limit value of duty (95%)</td>
</tr>
<tr>
<td></td>
<td>MTR_MIN_PWM_DUTY</td>
<td>(\frac{(((MTR_CARRIER_SET + 1) / 100) \times 5)}{100} - 1)</td>
<td>Min limit value of duty (5%)</td>
</tr>
<tr>
<td></td>
<td>MTR_SPEED_LIMIT</td>
<td>33000</td>
<td>Over speed limit(electrical angle)[rpm]</td>
</tr>
<tr>
<td></td>
<td>MTR_OVERVOLTAGE_LIMIT</td>
<td>15</td>
<td>Over voltage limit[V]</td>
</tr>
<tr>
<td></td>
<td>MTR_TIMEOUT_CNT</td>
<td>20</td>
<td>Undetected time = MTR_TIMEOUT_CNT * 1[ms]</td>
</tr>
<tr>
<td></td>
<td>MTR_SHIFT_ADJUST</td>
<td>0</td>
<td>Constant for adjust conducting pattern change timing</td>
</tr>
<tr>
<td></td>
<td>MTR_RPM_CALC_BASE</td>
<td>(60 \times MTR_TAU1_FREQ / 6)</td>
<td>Constant for speed measurement: 60[sec] * TAU1 timer frequency[Hz] / 6(times)</td>
</tr>
<tr>
<td></td>
<td>MTR_OVERSIZE_LIMIT</td>
<td>38</td>
<td>Speed deviation minimum value</td>
</tr>
<tr>
<td></td>
<td>MTR_STOP_WAIT_CNT</td>
<td>200</td>
<td>TAU0 interrupt period(50[us]) * MTR_STOP_WAIT_CNT = motor stop waiting time</td>
</tr>
</tbody>
</table>
### Table 5-10 Macro definitions list (5/5)

<table>
<thead>
<tr>
<th>File name</th>
<th>Macro name</th>
<th>Definition value</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtr_sssns_less_120.h</td>
<td>MTR_PATTERN_ERROR</td>
<td>0</td>
<td>Conducting pattern</td>
</tr>
<tr>
<td></td>
<td>MTR_U_PWM_VN_ON</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_V_PWM_VN_ON</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_W_PWM_UN_ON</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_U_PWM_VN_ON</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_V_PWM_UN_ON</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_W_PWM_VN_ON</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MTR_CW</td>
<td>0</td>
<td>CW</td>
</tr>
<tr>
<td></td>
<td>MTR_CCW</td>
<td>1</td>
<td>CCW</td>
</tr>
<tr>
<td></td>
<td>MTR_AVG_NEW</td>
<td>(1 - MTR_AVG_OLD)</td>
<td>factor of weighted average</td>
</tr>
<tr>
<td></td>
<td>MTR_OVERSIZE_LIMIT</td>
<td>38</td>
<td>for oversize check</td>
</tr>
<tr>
<td></td>
<td>MTR_FLG_CLR</td>
<td>0</td>
<td>for flag clear</td>
</tr>
<tr>
<td></td>
<td>MTR_FLG_SET</td>
<td>1</td>
<td>for flag set</td>
</tr>
<tr>
<td></td>
<td>MTR_POLE_PAIR</td>
<td>4</td>
<td>pole pairs</td>
</tr>
<tr>
<td></td>
<td>MTR_INITIAL_MODE</td>
<td>0x00</td>
<td>Initialization mode</td>
</tr>
<tr>
<td></td>
<td>MTR_OPENLOOP_MODE</td>
<td>0x01</td>
<td>Open loop mode</td>
</tr>
<tr>
<td></td>
<td>MTR_BEMF_120_MODE</td>
<td>0x03</td>
<td>Sensorless operation mode</td>
</tr>
<tr>
<td></td>
<td>MTR_OVER_CURRENT_ERROR</td>
<td>1</td>
<td>Overcurrent error</td>
</tr>
<tr>
<td></td>
<td>MTR_OVER_VOLTAGE_ERROR</td>
<td>2</td>
<td>Overvoltage error</td>
</tr>
<tr>
<td></td>
<td>MTR_OVER_SPEED_ERROR</td>
<td>3</td>
<td>Rotation speed abnormality error</td>
</tr>
<tr>
<td></td>
<td>MTR_TIMEOUT_ERROR</td>
<td>4</td>
<td>Timeout error</td>
</tr>
<tr>
<td></td>
<td>MTR_BEMF_ERROR</td>
<td>7</td>
<td>Induced voltage pattern error</td>
</tr>
<tr>
<td></td>
<td>MTR_UNDER_VOLTAGE_ERROR</td>
<td>8</td>
<td>Under voltage error</td>
</tr>
<tr>
<td></td>
<td>MTR_SHORT_ERROR</td>
<td>9</td>
<td>Short error</td>
</tr>
<tr>
<td></td>
<td>MTR_UNKNOWN_ERROR</td>
<td>0xff</td>
<td>Undefined error</td>
</tr>
<tr>
<td></td>
<td>MTR_MODE_STOP</td>
<td>0</td>
<td>Stop status</td>
</tr>
<tr>
<td></td>
<td>MTR_MODE_RUN</td>
<td>1</td>
<td>Rotating status</td>
</tr>
<tr>
<td></td>
<td>MTR_MODE_ERROR</td>
<td>2</td>
<td>Error status</td>
</tr>
<tr>
<td></td>
<td>MTR_SIZE_STATE</td>
<td>3</td>
<td>Status count</td>
</tr>
<tr>
<td></td>
<td>MTR_EVENT_STOP</td>
<td>0</td>
<td>Motor stop event</td>
</tr>
<tr>
<td></td>
<td>MTR_EVENT_RUN</td>
<td>1</td>
<td>Motor startup event</td>
</tr>
<tr>
<td></td>
<td>MTR_EVENT_ERROR</td>
<td>2</td>
<td>Motor error event</td>
</tr>
<tr>
<td></td>
<td>MTR_EVENT_RESET</td>
<td>3</td>
<td>Motor reset event</td>
</tr>
<tr>
<td></td>
<td>MTR_SIZE_EVENT</td>
<td>4</td>
<td>Events count</td>
</tr>
</tbody>
</table>
5.7 Control flow (flow chart)

(1) Main process

- main process
  - Initialization of peripheral functions
  - Initialization of 'main' used variables
  - Pre-driver error is not occur?
    - NO: Initialization processing
    - YES: Initialization of sequence processing
  - Reset processing
  - User interface
  - watch Dog Clear
  - Error check

- User Interface
  - Getting motor state
  - Getting switch state
  - Calculating motor rotate speed
  - Checking motor state and mode processing
  - Reset motor rotate direction
  - Setting motor rotate speed
  - End
(2) Carrier cycle interruption process

Carrier interrupt

Zero-crossing determination

Waiting motor stop?

NO

Elapse motor stop waiting time?

NO

YES

Clear motor stop waiting flag

Clear motor stop waiting counter

Clear timer RD compare match flag

End
(3) 1 [ms] interruption process

1 [ms] interrupt

Error detection (Overvoltage, rotation speed, timeout)

Motor mode is 120 degrees control mode?

YES

Speed control counter increment

Openloop counter bigger than specified value?

YES

speed contori counter clear

Speed PI control

Open-loop mode?

NO

YES

Open-loop control

End of interrupt
(4) Pre-driver error process

[Flowchart diagram]

1. Pre driver error process
2. Detection ERR1 or ERR2 is Low level?
   - YES: Stopping motor control
     - Clear pulse output force off register
     - Control MUTE port
   - NO
3. End
(5) Overcurrent interrupts process

```
(1) Overcurrent error process
   ↓
Error event
   ↓
Update error status
   ↓
End
```
Website and Support

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

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

All trademark and registered trademarks are the property of their respective owners.
## Revision History

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Data</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Mar.31.2017</td>
<td>—</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>


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 disputes 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, drawing, chart, program, algorithm, 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 otherwise misappropriate 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, copy or otherwise misappropriation of Renesas Electronics products.

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 depend 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.

6. Renesas Electronics products are neither intended nor 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 damages (space and underwater 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 third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.

7. When using the Renesas Electronics products, refer to the latest product information (data sheets, users' 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 radiation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions or failure or accident arising out of the use of Renesas Electronics products beyond such specified ranges.

8. 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 usage conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please ensure to implement safety measures to guard them against the possibility of bodily injury, injury or damage caused by fire, and social damage in the event of 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 by your own responsibility as warranty for your products/system. Because the evaluation of microcomputer software alone is very difficult and not practical, please evaluate the safety of the final products or systems manufactured by you.

9. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please investigate applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive carefully and sufficiently and use Renesas Electronics products in compliance with all 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.

10. 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 not use Renesas Electronics products or technologies for (1) any purpose relating to the development, design, manufacture, use, stockpiling, etc., of weapons of mass destruction, such as nuclear weapons, chemical weapons, or biological weapons, or missiles (including unmanned aerial vehicles (UAVs)) for delivering such weapons, (2) any purpose relating to the development, design, manufacture, or use of conventional weapons, or (3) any other purpose of disturbing international peace and security, and you shall not sell, export, lease, transfer, or release Renesas Electronics products or technologies to any third party whether directly or indirectly with knowledge or reason to know that such third party will engage in the activities described above. When exporting, selling, transferring, etc., Renesas Electronics products or technologies, you shall comply with any applicable export control laws and regulations promulgated and administered by the governments of the countries asserting jurisdiction over the parties or transactions.

11. This document shall not be reproduced, reprinted 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. (Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries. (Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.