RL78/F24

Sensorless Trapezoidal Control of BLDC Motor by MCU

R01AN6308EJ0110
Rev.1.10
2023.06.30

Introduction

This application note intends to describe a sample program for operating the 3-phase brushless DC motor with sensorless trapezoidal control method, by using the functions of RL78/F24 product.

The sample program is only for your reference and Renesas Electronics Corporation never guarantee the operations.

Please use this sample program after carrying out a thorough evaluation in a suitable environment.

Target Device

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

- RL78/F24 (R7F124FGJ)

Contents

1. Overview

This application note describes an example that RL78/F24 product executes a speed control of brushless DC motor (here in after referred to as BLDC motor) by using is driven sensorless trapezoidal control method.

1.1 Usage of the System

This system (sample program) enables trapezoidal control by using an RL78/F24 micro controller mounted CPU board, an inverter board for motor control “RTK7F124FGS00000BJ” and BLDC motor “TG-55N-KA”.

Remark

• TG-55N-KA: BLDC motor (Product of Tsukasa Electric CO., LTD.) <https://www.tsukasa-d.co.jp/>

1.2 Development Environment

Table 1-1 and Table 1-2 show the sample program development environment covered by this application note.

Table 1-1. Software Development Environment

Renesas CS+		
	IDE Version	CS+ for CC E8.07.00 [01 Dec 2021]
	Compiler	CC-RL E1.11.00
IAR		
	IDE Version	IAR Embedded Workbench IDE 8.5.2.7561 (8.5.2.7561)
	Compiler	IAR C/C++ Compiler for Renesas RL78 4.21.3.2447 (4.21.3.2447)

Table 1-2. Hardware Development Environment

On-chip Debugging Emulator	E2 emulator Lite
	E2 emulator
MCU Part Name	RL78/F24 (R7F124FGJ)
Inverter Board	RTK7F124FGS00000BJ
BLDC Motor	PMSM (TG-55N-KA)

2. System Overview

2.1 Hardware Configuration

The hardware configuration is shown in Figure 2-1.

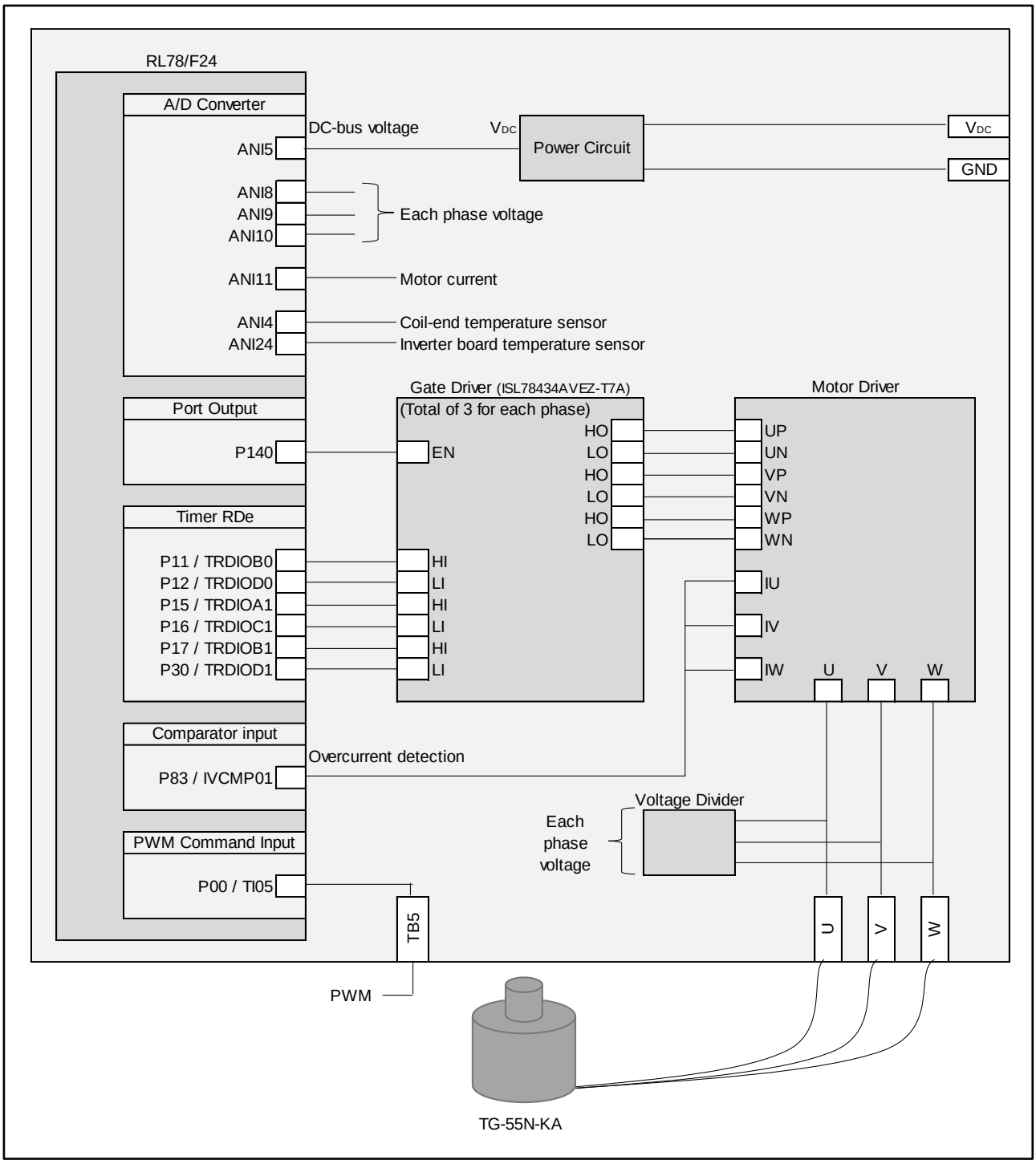

Figure 2-1 Hardware Configuration Diagram

2.2 Hardware Specifications

2.2.1 Hardware Interface

Table 2-1 shows a list of hardware interfaces for this system.

Table 2-1 Hardware Interface

RL78/F24 Pin Name	Function
P86 / ANI8	U-phase voltage measurement
P87 / ANI9	V-phase voltage measurement
P90 / ANI10	W-phase voltage measurement
P85 / ANI5	DC-bus voltage (V_{dc}) measurement
P91 / ANI11	Motor current (I_{dc}) measurement
P84 / ANI4	Coil-end temperature measurement
P125 / ANI24	Inverter board temperature measurement
P11 / TRDIOB0	Complementary PWM (U_P) or port output
P12 / TRDIOD0	Complementary PWM (U_N) or port output
P15 / TRDIOA1	Complementary PWM (V_P) or port output
P17 / TRDIOB1	Complementary PWM (W_P) or port output
P16 / TRDIOC1	Complementary PWM (V_N) or port output
P30 / TRDIOD1	Complementary PWM (W_N) or port output
P140	Gate driver EN signal output
P83 / IVCMP01	Comparator input for overcurrent detection
P00 / TI05	PWM input

2.2.2 Peripheral Functions

Table 2-2 shows the peripheral functions of RL78/F24 product used in this system.
For details, see “4. Description of Peripheral Functions Used”.

Table 2-2 List of Peripheral Functions Used in This System

Peripheral Function	Usage
A/D Converter	• U-, V-, and W-phase voltage measurement • DC-bus voltage (V_{dc}) measurement • Motor current (I_{dc}) measurement • Temperature measurement on the motor coil-end • Temperature measurement on the PCB
Timer RDe	• PWM output using extended complementary PWM mode (Normal phase: 3 ch, Counter phase: 3 ch) • PWM output forced cutoff (using PWMOPA)
Port	• Motor control signal output • Gate driver EN signal output
Timer Array Unit (TAU)	• 1 [ms] interval timer • PWM input
D/A Converter	Generating threshold voltage for internal comparator
Comparator	For overcurrent detection

2.3 Software Configuration

2.3.1 File Configuration

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

Table 2-3 Folders and Files Configuration of Sample Program

RL78F24_120_ADC_V105		
prj	RL78F24_120_ADC.mtpj	CS+ Project File (for CS+)
	RL78F24_120_ADC.dep	IAR workspace files (for IAR)
	RL78F24_120_ADC.ewd	
	RL78F24_120_ADC.ewp	
	RL78F24_120_ADC.ewt	
	RL78F24_120_ADC.eww	
inc	iodefine.h	SFR definition file (for CC-RL)
	mtr_cpu_setting.h	CPU processing header
	mtr_fix_calc.h	Fixed-point processing header
	mtr_function.h	Common processing header
	mtr_interrupt_bemf.h	Motor control interruption header
	mtr_inv_setting.h	Inverter processing header
	r_compiler_common.h	Compiler processing header
	r_mtr_sequence_api.h	State control processing header
	r_mtr_user_control_api.h	User API header
	r_mtr_user_control_cfg.h	User configuration header
	r_typedefs.h	Type declaration header
src	mtr_main.c	Main function
	mtr_cpu_setting.c	CPU processing
	mtr_fix_calc.c	Fixed-point processing
	mtr_function.c	Common processing
	mtr_interrupt_bemf.c	Motor control interruption handler
	mtr_inv_setting.c	Inverter processing
	r_mtr_sequence_api.c	State control processing
	r_mtr_user_control_api.c	User processing
	mcu_debug_option_for_iar.c	Option byte setting (for IAR)
asm	cstart.asm	Startup (for CC-RL)
	hwinit.asm	Hardware initialization (for CC-RL)
	stkinit.asm	Stack pointer initialization (for CC-RL)
	f24opt.asm	Option byte setting (supplement) (for CC-RL)

2.3.2 Module Configuration

Module configuration of the sample program is described below.

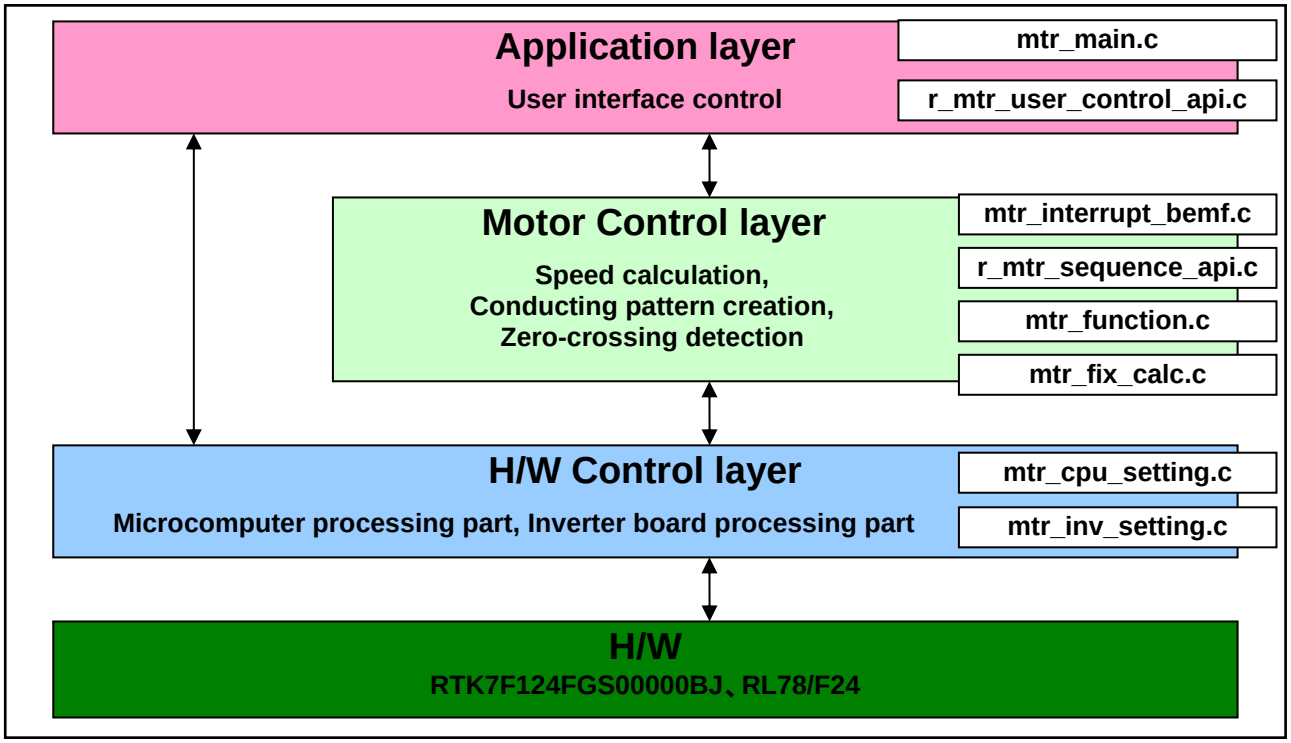

Figure 2-2 Module Configuration of Sample Program

2.4 Software Specifications

Table 2-4 shows a basic specifications of sample program.

Table 2-4 Basic Specifications of Sample Program

Item	Content
Control method	trapezoidal control method
Motor rotation start / stop	Select one of the following two to rotate the motor. • After the reset is released, the rotation control of the motor starts. – After 3 [s], operation starts with 1000 [rpm] as the rotation speed command value. – 10 [s] after the above operation, add 500 [rpm] rotation speed command values every 10 [s] until the speed command value reaches 3000 [rpm]. – After the rotation speed command value reaches 3000 [rpm], the speed command value is subtracted by 500 [rpm] every 10 [s]. – After the rotation speed command value reaches 0 [rpm], the motor will stop rotating. • After the reset is released, the rotation speed command value is controlled by the PWM signal input. – Input PWM signal is 10 [Hz] to 1000 [Hz]. – When the duty of the input PWM signal is 100%, the rotation speed command value is set to 3000 [rpm]. – When the duty of the input PWM signal is 0%, the rotation speed command value is set to 0 [rpm].
Position detection of rotor magnetic pole	Using A/D converter, the position is detected by the induced voltage every 60-degree.
PWM carrier frequency	20 [kHz]
Control cycle	• Executes zero-crossing detection from the induced voltage per carrier cycle • Determination of PWM duty setting and conducting pattern.
Rotation speed control range	Both CW/CCW are supported from 500 [rpm] to 3000 [rpm].
Rotation speed operation	In conducting pattern change, calculates rotate speed from elapse time of previous one.
Speed control (Speed PI control)	Obtains the speed command value form rotation speed command value setting function, and performs speed control by PI control (10 [ms] cycle).
Processing stop for protection	Deactivates the motor control signal output (six outputs) if any of the following conditions are met. – When the motor drive (DC-bus) voltage exceeds 28.0 [V] or becomes less than 8.0 [V] (monitored for each carrier interruption handler). – When the bus current (I_{dc}) exceeds 10.0 [A] (monitored for each carrier interruption handler). – Rotation speed exceeds 10000 [rpm] (monitored for each 1 [ms]). – No zero-crossing detected for 200 [ms] in sensorless drive mode. – When overcurrent due to hardware is detected. – When the inverter board temperature exceeds 125 [°C] (monitored for each 1 [ms]). – When the coil-end temperature exceeds 180 [°C] (monitored for each 1 [ms]).

3. Motor Control Method

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

3.1 Sensorless trapezoidal 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 a closed circuit is proportion to the time rate of change of the magnetic flux through the circuit.

For example, considers 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.)

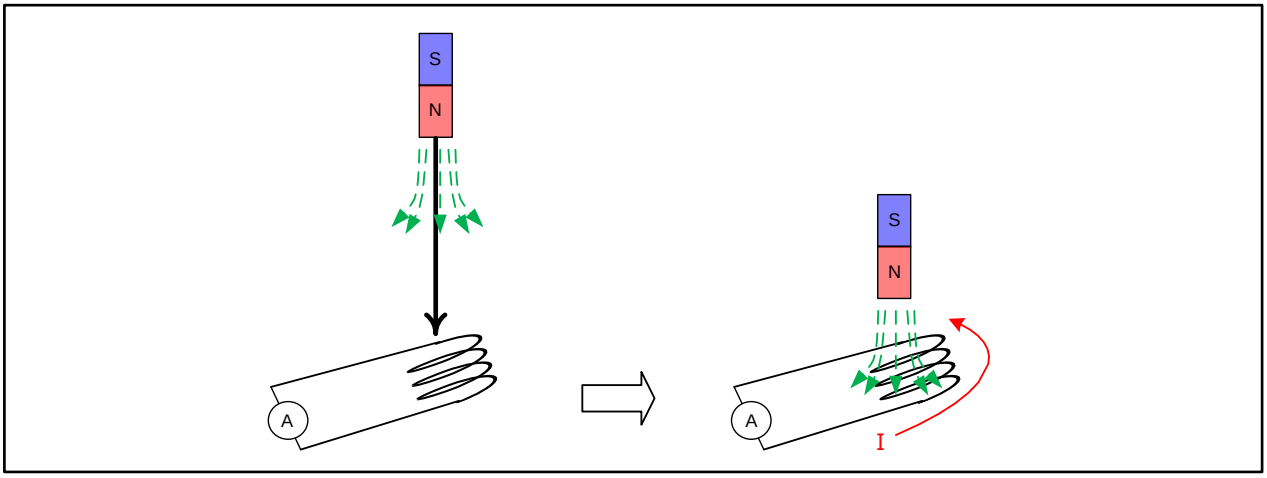

Figure 3-1 Induced Voltage Depending on the Coil Magnet

This induced voltage E_m is expressed by the magnetic flux φ_m as the following formula.

$$E_m = \frac{d}{dt}\varphi_m \ldots \text{ formula (1)}$$

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.

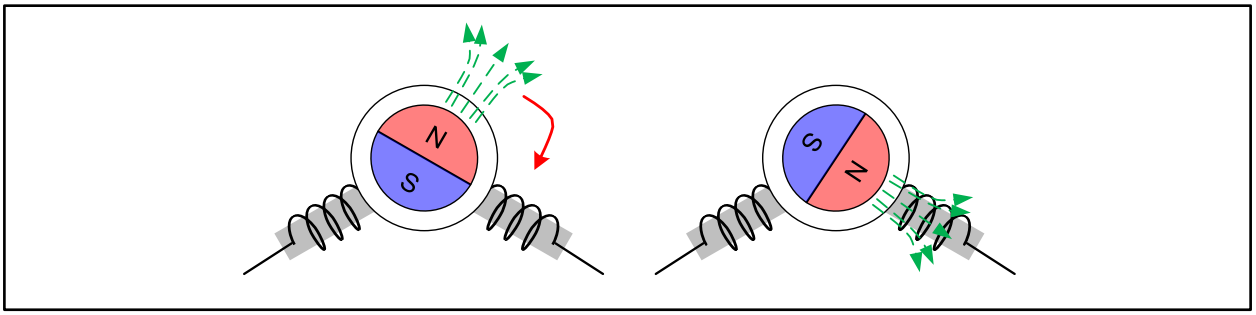

Figure 3-2 Induced Voltage in the Rotating Permanent Magnetic Synchronous Motor

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 'θ = 0'.

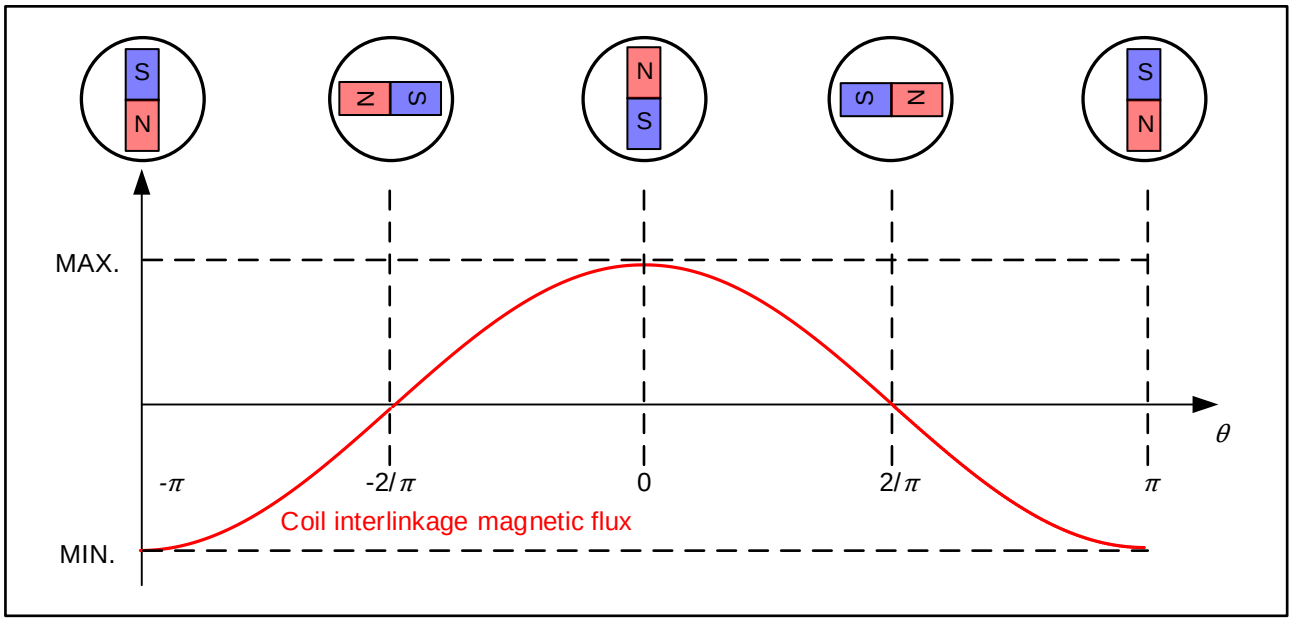

Figure 3-3 Fluctuation of Interlinkage Magnetic Flux

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

If considered similarly for V-phase and W-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.

$$\varphi_u = \varphi_m \cos\theta$$
$$\varphi_v = \varphi_m \cos(\theta - \frac{2}{3}\pi)$$
$$\varphi_w = \varphi_m \cos(\theta - \frac{4}{3}\pi)$$

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

$$E_u = \frac{d}{dt}\varphi_u = \frac{d}{dt}\varphi_m \cos\theta = -\omega\varphi_m \sin\theta = \omega\varphi_m \cos(\theta + \frac{\pi}{2})$$
$$E_v = \frac{d}{dt}\varphi_v = \frac{d}{dt}\varphi_m \cos(\theta - \frac{2}{3}\pi) = -\omega\varphi_m \sin(\theta - \frac{2}{3}\pi) = \omega\varphi_m \cos(\theta - \frac{\pi}{6})$$
$$E_w = \frac{d}{dt}\varphi_w = \frac{d}{dt}\varphi_m \cos(\theta - \frac{4}{3}\pi) = -\omega\varphi_m \sin(\theta - \frac{4}{3}\pi) = \omega\varphi_m \cos(\theta - \frac{5}{6}\pi)$$

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.

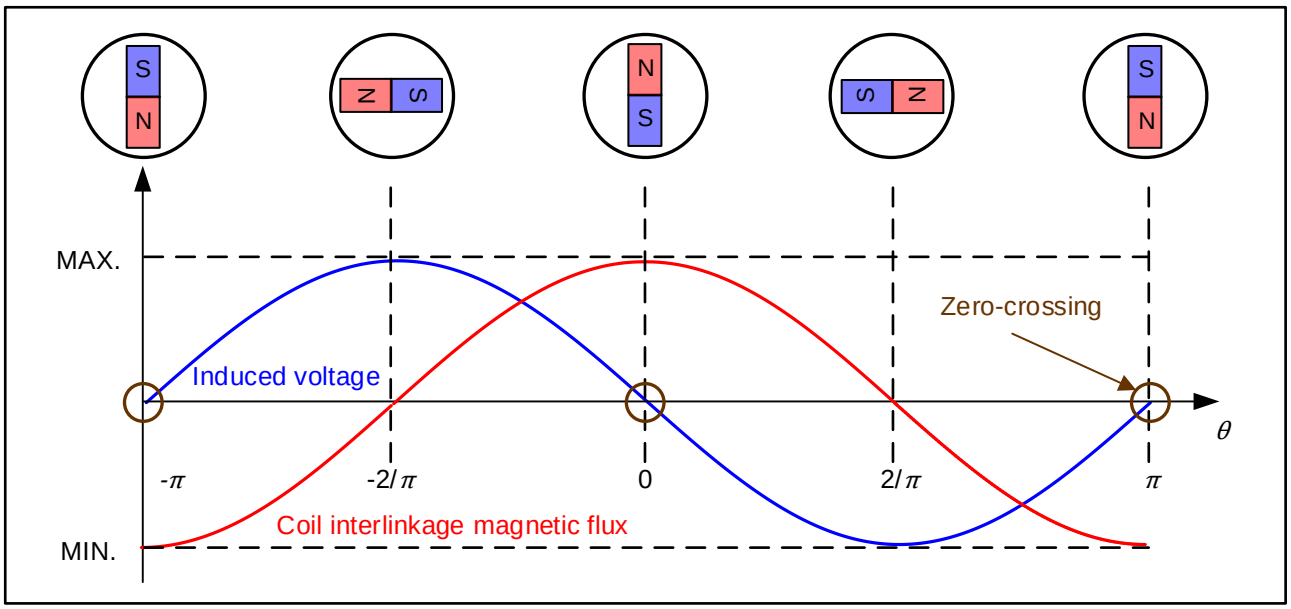

Figure 3-4 Zero-crossing of the Induced Voltage

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

During the driving in 120-degree 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 non-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-degree can be detected by this process in the same way as resolution of hall sensor.

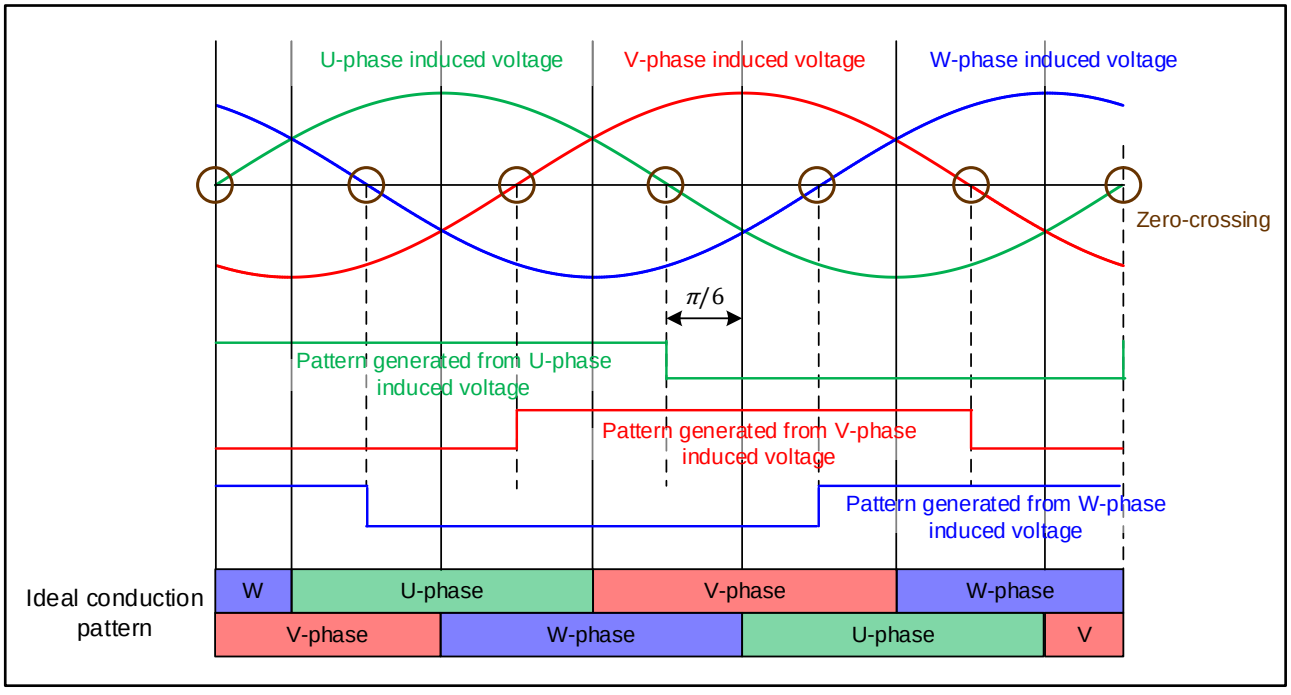

Figure 3-5 Relation between Conducting Pattern and Zero-crossing (Upper Arm Chopping)

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 $\pi/6$ 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 $\pi/6$ 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.

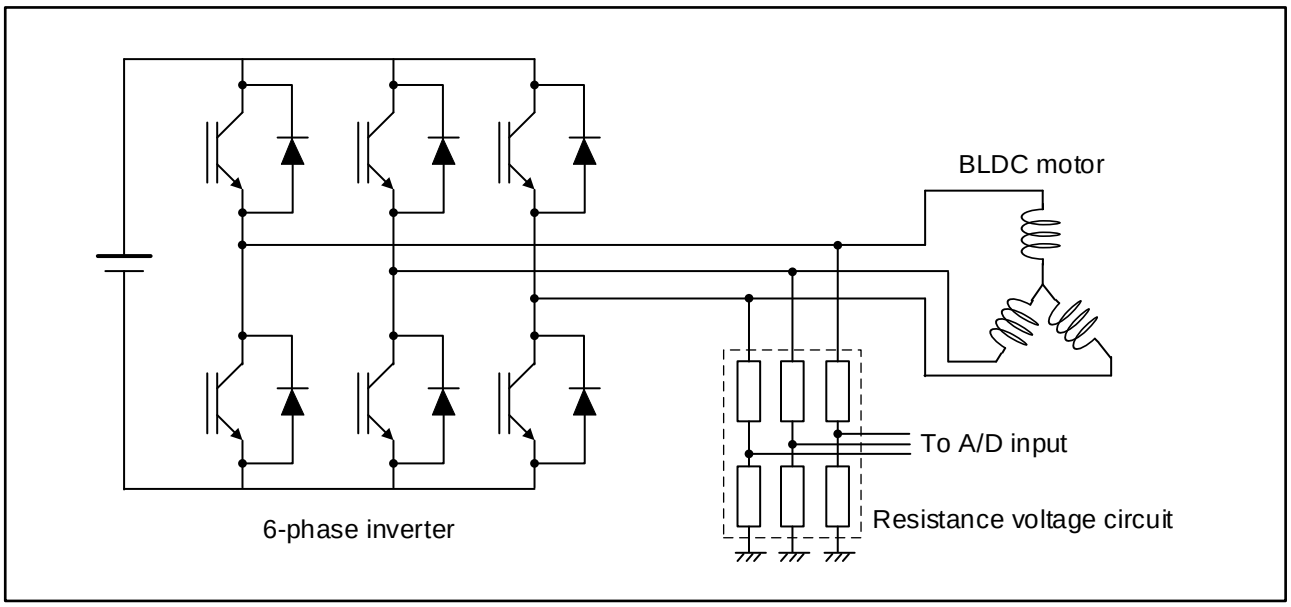

Figure 3-6 Comparator Less Method

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.

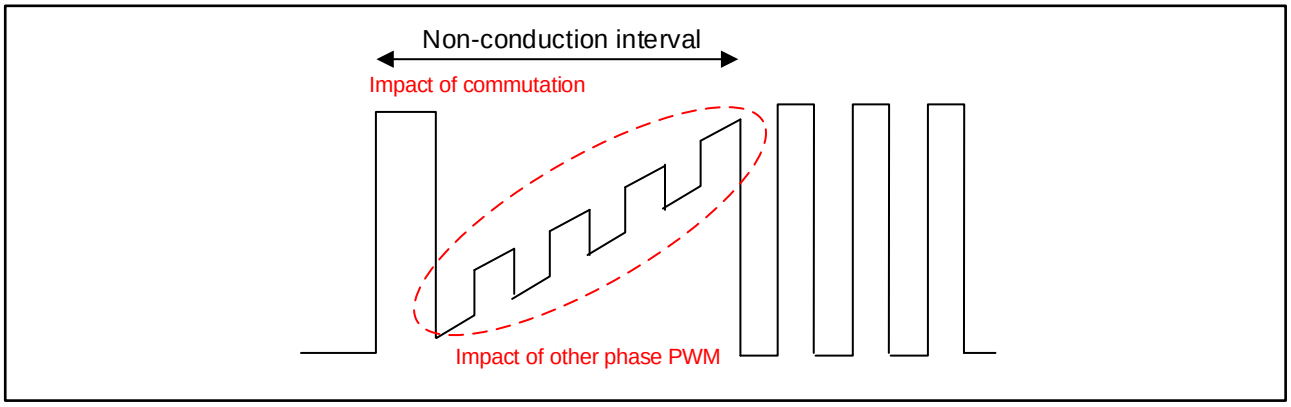

Figure 3-7 Overview Diagram of Impact of the Commutation and Other Phase PWM

In this system, impact is removed by using the simple filter route and the software.

The following software is specifically used to remove the commutation voltage.

The first excludes the effect of commutation voltage by disabling zero-crossing detection after the start of zero-crossing detection. At this time, the zero-crossing detection function for each carrier cycle is disabled for the number of times that zero-crossing detection is disabled.

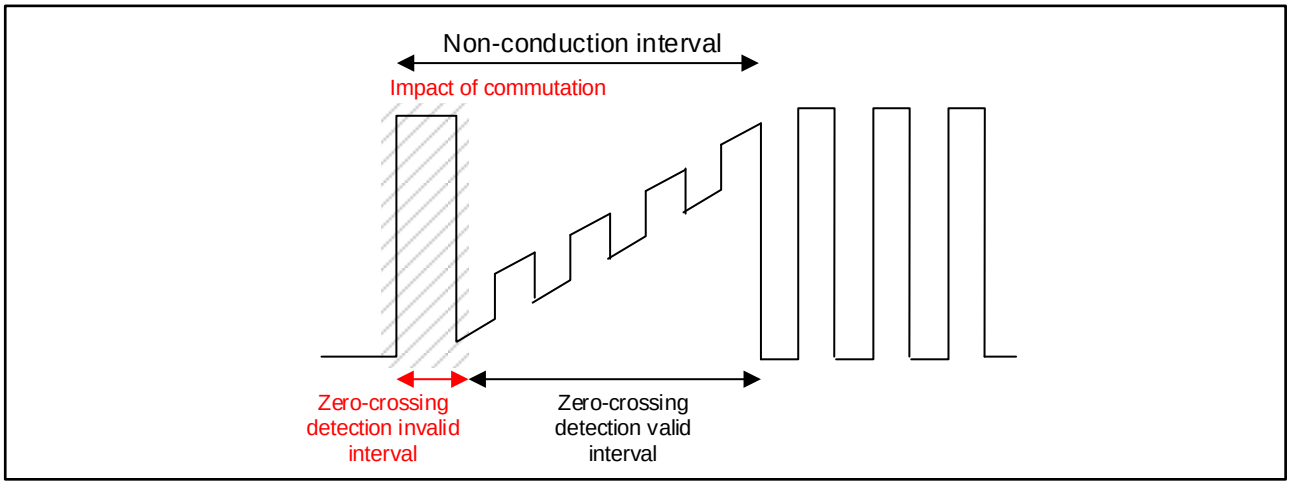

Figure 3-8 Setting the Zero-crossing Detection Invalid Period

Table 3-1 Zero-crossing Detection Invalid Period Parameter

Parameter Name	Initial Value	Contents
MTR_BEMF_DETECT_GUARD_CNT	2	Zero-crossing detection invalid count setting

Next, sets the range of zero-crossing detection voltage. If an induced voltage exceeding the set range is detected, the zero-crossing detection will be invalid.

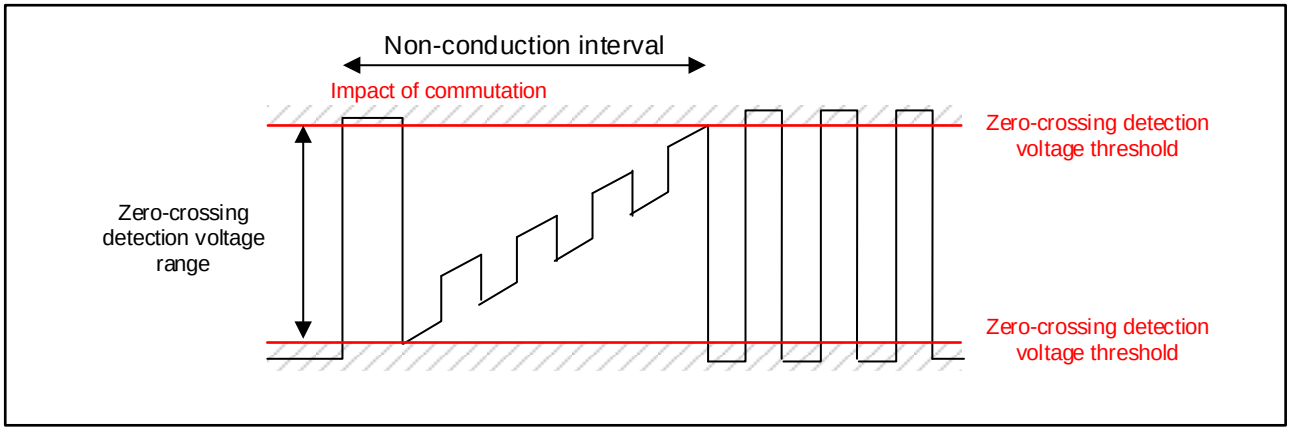

Figure 3-9 Zero-crossing Detection Voltage Range Setting

Table 3-2 Zero-crossing Detection Voltage Range Parameter

Parameter Name	Initial Value	Contents
USER_CFG_BEMF_DETECT_END_TH	30	Zero-crossing detection voltage threshold [A/D value]

3.3 Start-up Method

This sensorless trapezoidal control uses the induced voltage due to the change in the magnetic flux of the magnet (rotor) to estimate the position of the magnetic pole at every 60-degree of electrical angle.

Induced voltage does 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.

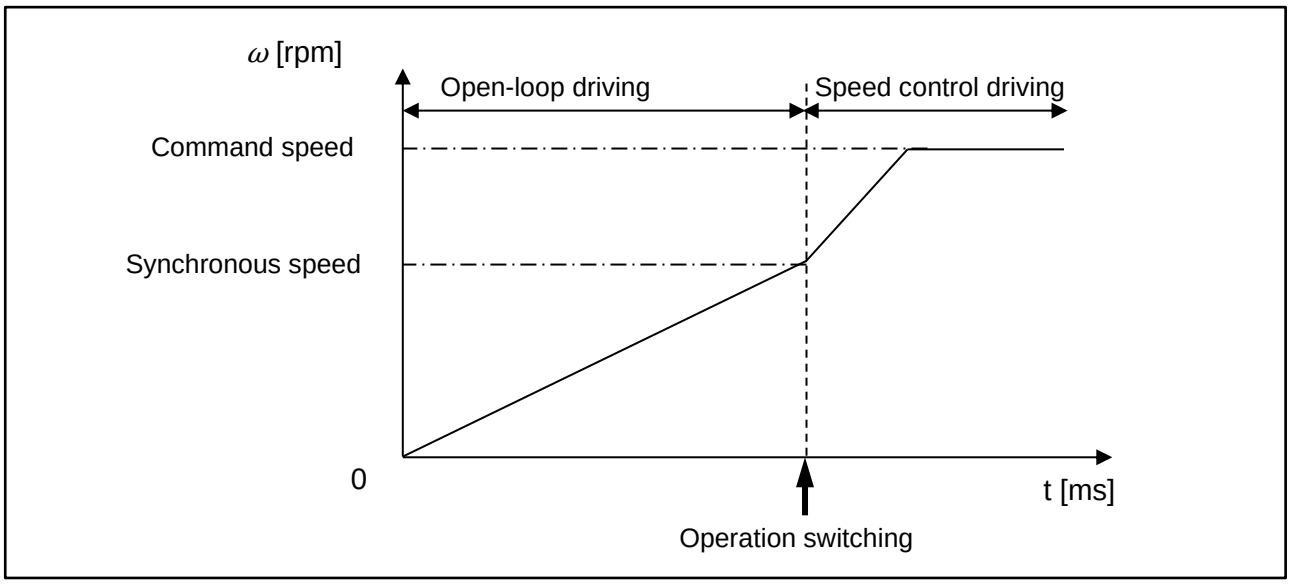

Figure 3-10 Diagram of Start-up Operation

3.4 Position Estimate Operation

This system has estimated angle information inside the sample program. The estimated angle data is integrated by calculating the increment angle for each carrier cycle from the rotation speed. The conducting pattern is switched when the calculated estimated angle exceeds the pattern switching angle. Also, when zero-crossing is detected, the estimated angle is corrected to the zero-crossing detection angle.

The correction value of the estimated angle is the value obtained by adding the increase angle for two carrier cycles to the zero-crossing detection angle in consideration of the delay for two carrier cycles. The causes of the delay for two carrier cycles are shown below.

- To prevent false detections, zero-crossing detection is determined when zero-crossing is detected twice in a row. Therefore, there is a delay of one cycle in the carrier cycle from the time when the induced voltage reaches the zero-crossing detection voltage until the internal program determines that the detection is completed.
- The process of switching the conducting pattern is performed by the interrupt handler of the next carrier cycle after the estimated angle detects the pattern switching angle. Therefore, there will be a delay of one carrier cycle.

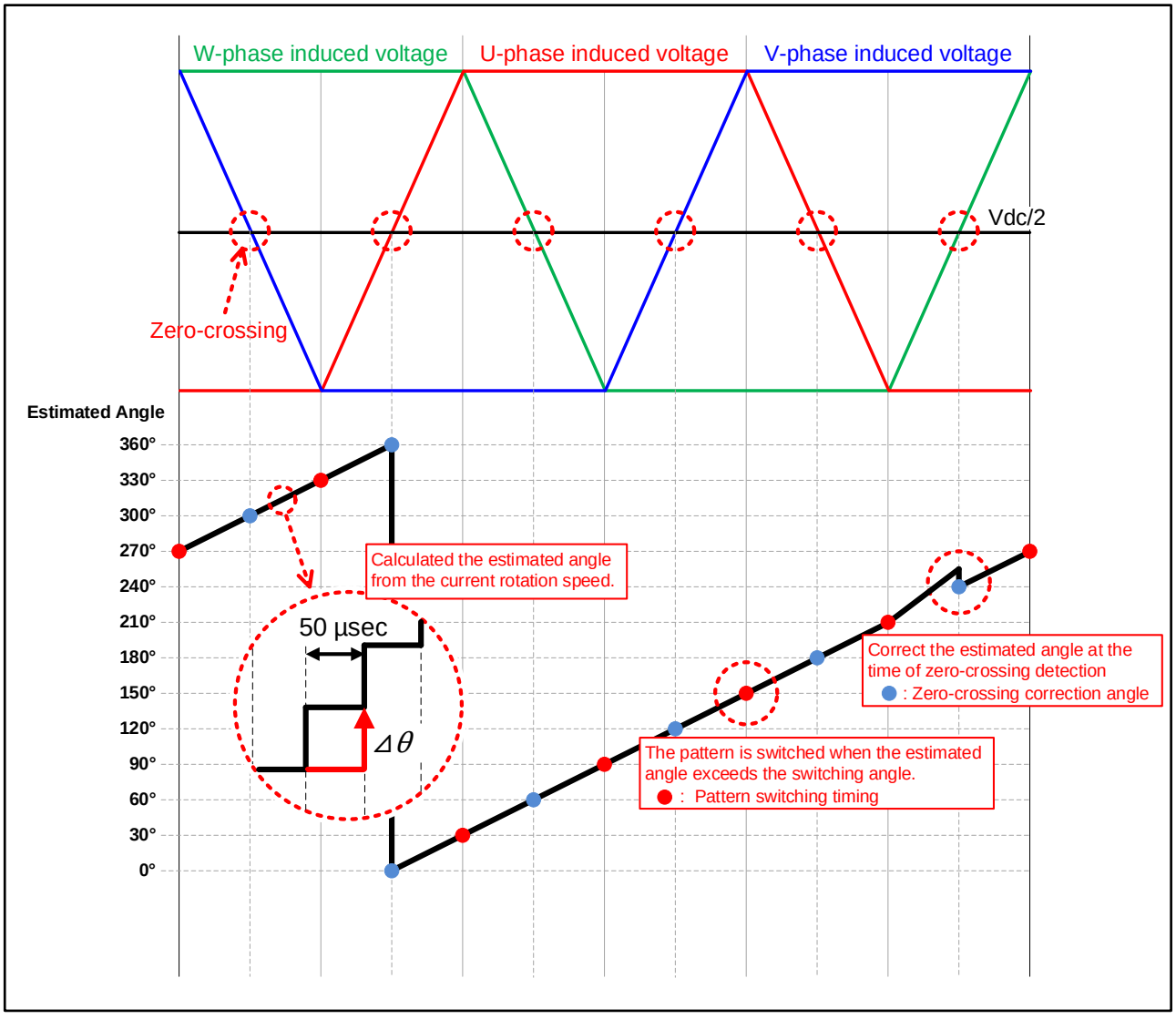

Figure 3-11 Virtual Hall Sensor Pattern (Upper Arm Chopping)

3.5 Rotation Speed Control

In this system, the number of interrupt occurrences is counted as a global pulse number in the PWM carrier cycle interrupt processing. Since PWM carrier interrupts occur every PWM cycle, use this global pulse number to measure the time for any period.

In the rotation speed control process of this system, the elapsed time for each pattern switching for 6 times is obtained from the global pulse number, and the rotation speed is calculated from this elapsed time.

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

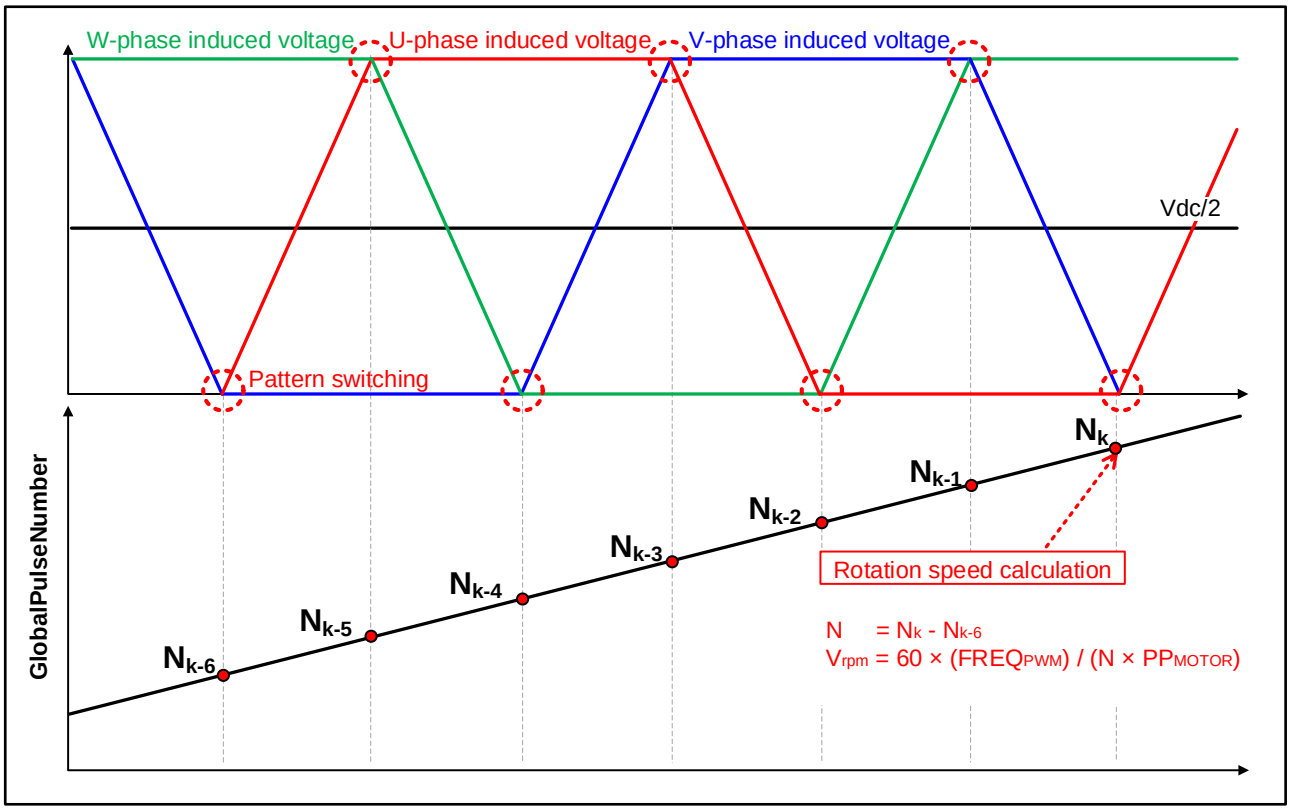

Figure 3-12 Rotation Speed Calculation Method

Table 3-3 Motor Rotation Speed Calculation Parameters

Item	Variable / Macro Name	Contents
V_{rpm}	g_mtr_rpm_speed	Rotation speed after LPF [rpm]
FREQPWM	TRDE_CARRIER_FREQUENCY_HZ	PWM carrier frequency [Hz]
N	–	Number of PWM pulses when rotated 360-deg at electrical angle
PPMOTOR	USER_CFG_MOTOR_POLE_PAIRS	Motor pole logarithm
N_k	g_mtr_global_pulse_number	Current global pulse number
N_{k-6}	g_mtr_global_pulse_number_old	Global pulse number when calculating rotation speed

This system is using PI control for motor rotation speed control. The speed PI control is performed every 10 [ms] by 1 [ms] interval timer interrupt processing. This speed PI control interval can be changed by a parameter (g_mtr_speed_pi_interval). The PWM command value is calculated by the motor rotation speed PI by the following formula.

$$Nerr_k = N^* - N_{rpm}$$
$$d^*_k = d^*_{k-1} + K_P \times (Nerr_k - Nerr_{k-1}) + K_I \times err_k$$

Table 3-4 Speed PI Control Parameters and Variables

Item	Variable / Macro Name	Fixed-point number format	Contents
$Nerr_k$	–	–	Speed deviation [rpm]
N^*	g_mtr_rpm_ref	–	Speed command value [rpm]
N_{rpm}	g_mtr_rpm_speed	–	Current speed value [rpm]
d^*_k	g_mtr_next_duty	Q14	PWM command value
K_P	g_mtr_speed_kp_factor	Q14	Proportional coefficient
K_I	g_mtr_speed_ki_factor	Q14	Integral coefficient

This system uses PWM control to control the output voltage. PWM control is a control method that adjusts the average voltage by changing the duty of the PWM waveform (see Figure 3-13).

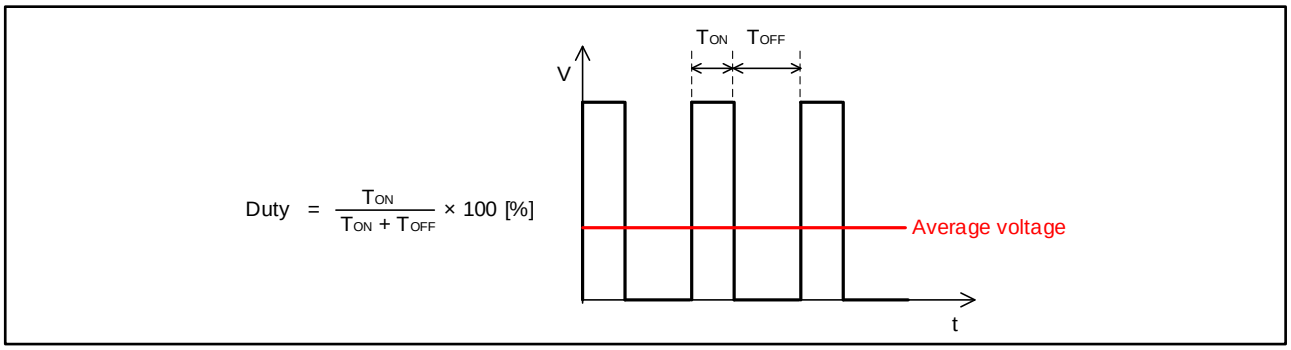

Figure 3-13 PWM Control

In this system, complementary PWM chopping (120-degree) 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 3-14.

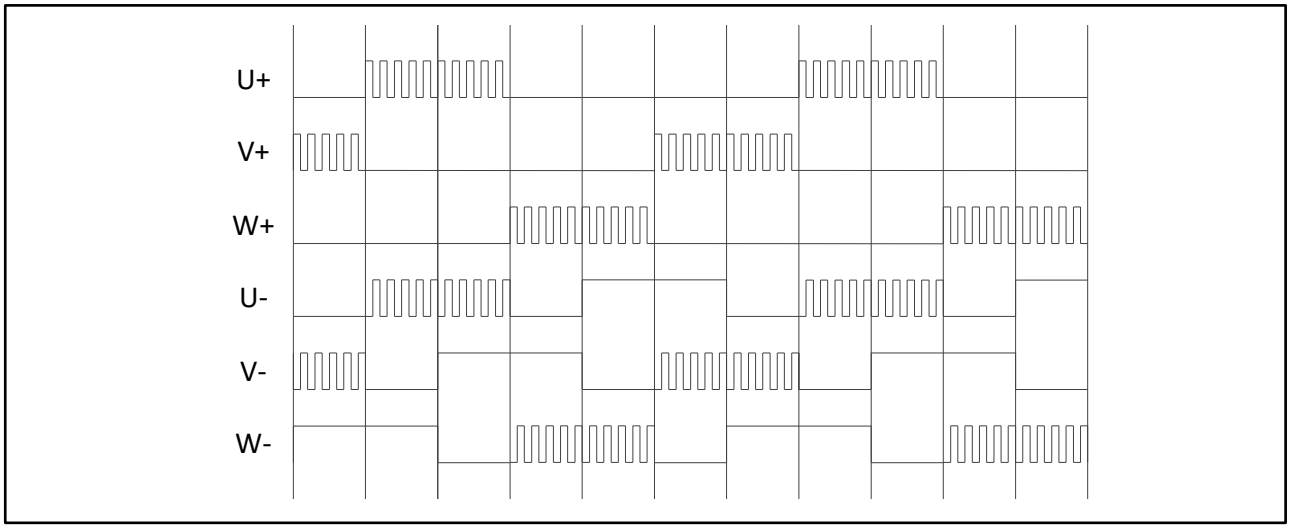

Figure 3-14 Waveform of Complementary PWM Chopping (120-degree)

3.6 Expression of Degree Value

In this system, the degree value is represented by a normalized 14-bit fixed-point value.

Table 3-5 List of Normalized Degree Values

Degree Measures	Circular Measures	Normalized Degree Values (Q14)
0 (360)	0	0
60	1.047197551	2731
120	2.094395102	5462
180	3.141592654	8192
240	4.188790205	10923
300	5.235987756	13654
359	6.265732015	16338

4. Description of Peripheral Functions Used

Peripheral functions used in this system are explained.

Following peripheral functions are explained in this chapter.

- A/D Converter
- Timer Array Unit (TAU)
- Timer RDe

4.1 A/D Converter Function

The A/D converter converts the analog input voltage to digital value. The target microcontroller (RL78/F24) incorporates one circuit of 12-bit A/D converter. Analog input of 19 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 4-1 and Table 4-2.

Table 4-1 Usage of A/D Converter (Obtained by Interrupt of INTTM01)

Channel	Item	Physical quantity per bit of A/D converted value
ANI4	Coil-end temperature measurement	Refer to Table 4-4.
ANI24	Inverter board temperature measurement	Refer to Table 4-5.

Remark INTTM01: Timer array unit 0 channel.1 interrupt

Table 4-2 Usage of A/D Converter (Obtained by Interrupt of INTTRD_ADTRG)

Channel	Item	Physical quantity per bit of A/D converted value
ANI5	DC-bus voltage (V_{dc}) measurement	65.0 [V] / 4095 = 0.0159 [V]
ANI8	U-phase voltage measurement	25.0 [V] / 4095 = 0.0061 [V]
ANI9	V-phase voltage measurement	25.0 [V] / 4095 = 0.0061 [V]
ANI10	W-phase voltage measurement	25.0 [V] / 4095 = 0.0061 [V]
ANI11	Motor current (I_{dc}) measurement	50.0 [A] / 4095 = 0.0122 [A]

Remark INTTRD_ADTRG: Timer RDe A/D conversion trigger

Table 4-3 List of A/D Conversion Target

Obtained target	Variable Name	Fixed-point number format	A/D Input Pin
DC-bus voltage (V_{dc})	g_mtr_vdc_value	Q8	ANI5
U, V, W-phase voltage	float_voltage	Q8	ANI8, ANI9, ANI10
Motor current (I_{dc})	g_mtr_idc_value	Q10	ANI11
Coil-end temperature measurement	g_th_cur_temp_motor	–	ANI4
Inverter board temperature	g_th_cur_temp_board	–	ANI24

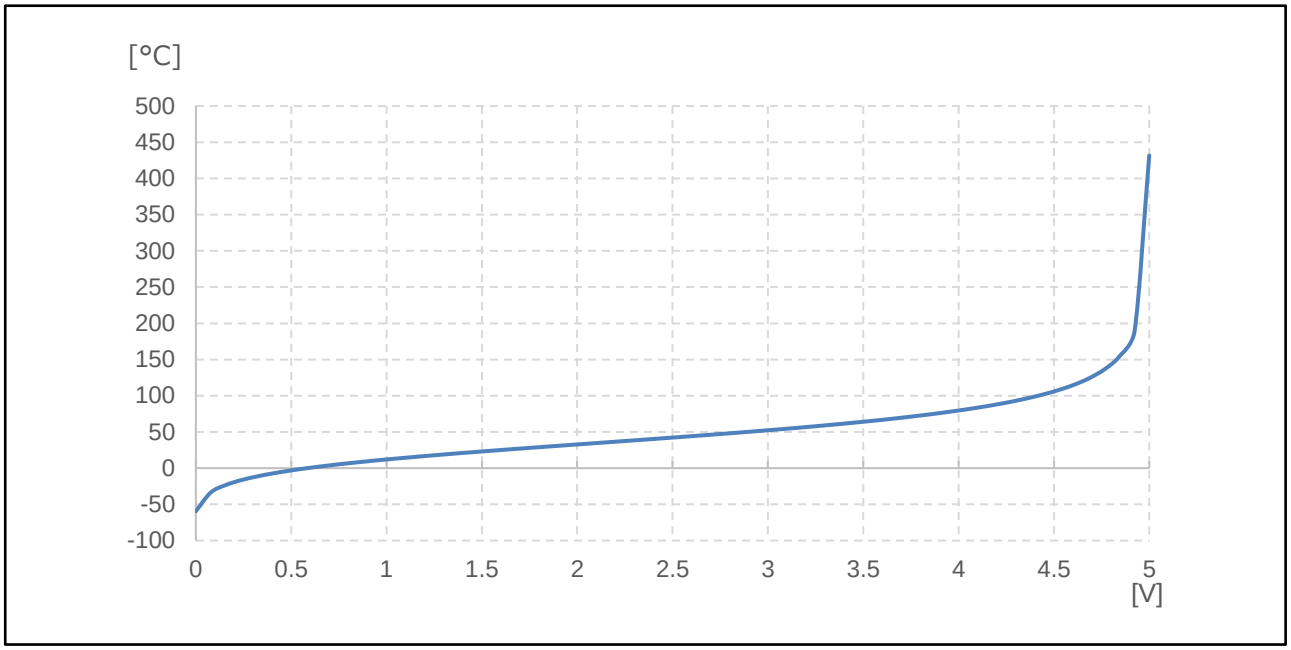

Figure 4-1 Coil-end Temperature Measurement Graph

Table 4-4 Coil-end Temperature Measurement Table

Voltage	Temp.	Voltage	Temp.	Voltage	Temp.	Voltage	Temp.
0.000	-59.393	1.328	19.469	2.657	45.264	3.985	79.031
0.078	-33.636	1.407	21.111	2.735	46.818	4.063	82.049
0.156	-23.353	1.485	22.716	2.813	48.396	4.142	85.326
0.234	-16.808	1.563	24.289	2.891	50.002	4.220	88.917
0.313	-11.870	1.641	25.836	2.969	51.641	4.298	92.896
0.391	-7.838	1.719	27.362	3.048	53.317	4.376	97.367
0.469	-4.391	1.797	28.870	3.126	55.035	4.454	102.478
0.547	-1.354	1.875	30.364	3.204	56.801	4.532	108.450
0.625	1.381	1.954	31.849	3.282	58.620	4.611	115.640
0.703	3.884	2.032	33.326	3.360	60.501	4.689	124.664
0.781	6.205	2.110	34.799	3.438	62.450	4.767	136.738
0.860	8.378	2.188	36.272	3.516	64.477	4.845	154.778
0.938	10.430	2.266	37.748	3.595	66.592	4.923	189.159
1.016	12.382	2.344	39.228	3.673	68.808	5.000	431.619
1.094	14.250	2.422	40.717	3.751	71.141		
1.172	16.047	2.501	42.217	3.829	73.607		
1.250	17.783	2.579	43.732	3.907	76.228		

Remarks The units of the table are as follows.

Voltage: [V], Temperature: [°C]

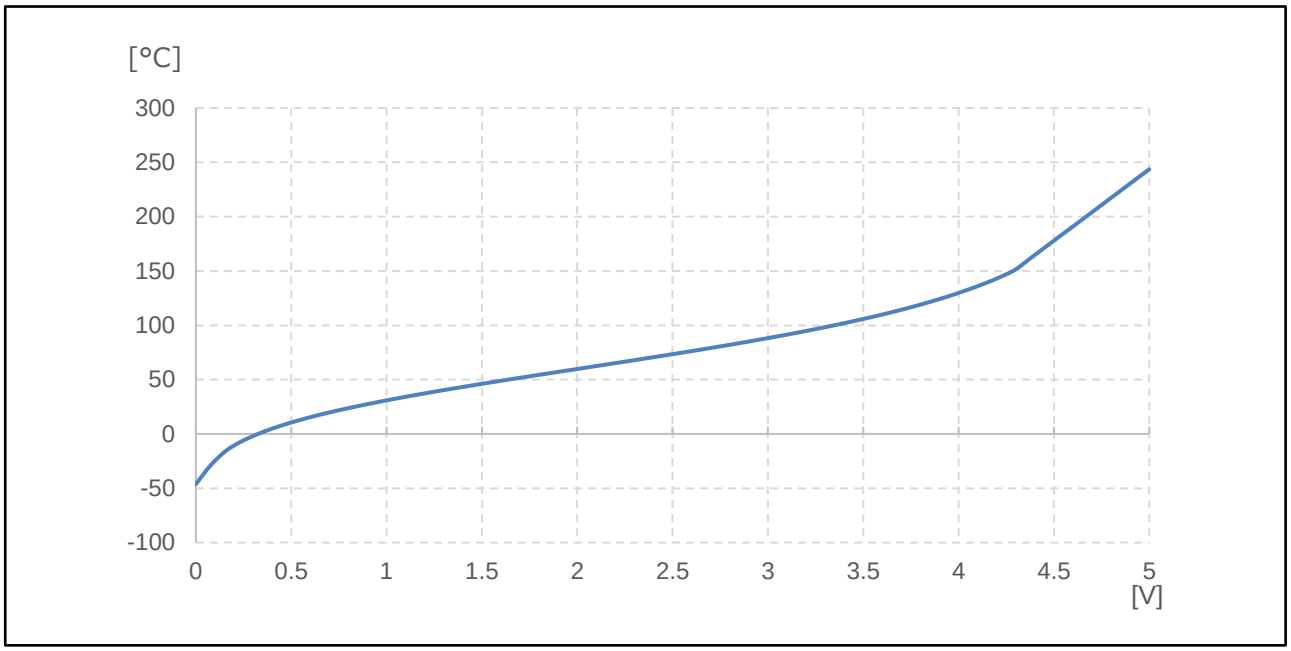

Figure 4-2 Inverter Board Temperature Measurement Graph

Table 4-5 Inverter Board Temperature Measurement Table

Voltage	Temp.	Voltage	Temp.	Voltage	Temp.	Voltage	Temp.
0.000	-46.154	1.328	41.174	2.657	77.864	3.985	128.909
0.078	-28.744	1.407	43.453	2.735	80.135	4.063	133.649
0.156	-15.757	1.485	45.688	2.813	82.45	4.142	138.83
0.234	-7.357	1.563	47.887	2.891	84.814	4.220	144.548
0.313	-0.95	1.641	50.055	2.969	87.233	4.298	151.285
0.391	4.326	1.719	52.2	3.048	89.716	4.376	161.566
0.469	8.869	1.797	54.326	3.126	92.27	4.454	171.848
0.547	12.898	1.875	56.44	3.204	94.905	4.532	182.129
0.625	16.546	1.954	58.545	3.282	97.629	4.611	192.41
0.703	19.903	2.032	60.647	3.360	100.456	4.689	202.691
0.781	23.029	2.110	62.75	3.438	103.397	4.767	212.973
0.860	25.969	2.188	64.858	3.516	106.468	4.845	223.254
0.938	28.758	2.266	66.975	3.595	109.688	4.923	233.535
1.016	31.42	2.344	69.107	3.673	113.076	5.000	243.656
1.094	33.978	2.422	71.257	3.751	116.659		
1.172	36.447	2.501	73.43	3.829	120.465		
1.250	38.842	2.579	75.631	3.907	124.533		

Remarks The units of the table are as follows.

Voltage: [V], Temperature: [°C]

4.2 Timer Array Unit (TAU) Function

Timer Array Unit (TAU) consists of eight 16-bit 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. The target microcontroller (RL78/F24) incorporates two units (16 channels in total).

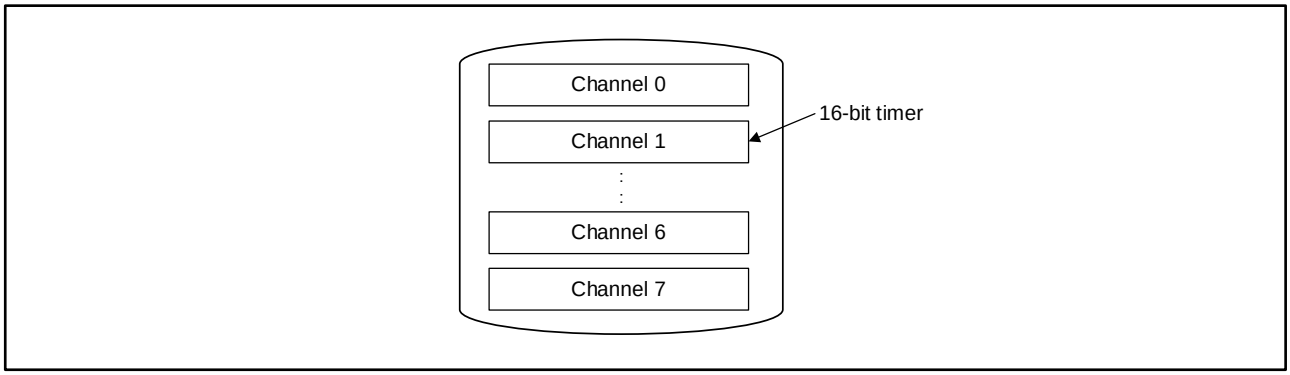

Figure 4-3 Timer Array Unit (TAU)

In this system, the Timer Array Unit is set as given in Table 4-6.

Table 4-6 Timer Array Unit (TAU) Setting Details

Unit	CH	Item	Contents	Usage
0	1	Operation mode of timer	Interval timer mode	Interval timer for 1 [ms]
		Source clock	CK00	
		Count clock frequency	40 [MHz]	
		Interruption cycle	1 [ms]	
		Setting value of timer data register (TDR01)	39999 ((1 [ms] / (1/40 [MHz])) - 1)	
	5	Operation mode of timer	Capture mode	For measuring the width of the input PWM signal
		Source clock	CK01	
		Count clock frequency	625 [kHz]	
		Interruption cycle	Both edge detection	
		Setting value of timer data register (TDR05)	–	
	6	Operation mode of timer	One-count mode	For input PWM signal timeout detection
		Source clock	CK01	
		Count clock frequency	625 [kHz]	
		Interruption cycle	100 [ms]	
		Setting value of timer data register (TDR06)	62499 ((100 [ms] / (1/625 [kHz])) - 1)	

4.3 Timer RDe Function

Timer RDe consists of two 16-bit timers (timer RD0 and timer RD1). In addition, the following 6 modes are provided for timer RDe function.

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

In this system, the timer RDe is set as given in Table 4-7.

Table 4-7 Timer RDe Setting Details

Item	Contents	Usage
Mode to use	Extended complementary PWM mode (Symmetric waveform output)	3-phase PWM output
PWM cycle	50 [µs]	
Dead time	1.0 [µs]	
Count source frequency	80 [MHz]	
Output level	Initial output is ‘Low’, Active level is ‘High’	
Buffer operation	Valid	
Pulse output forced cutoff	Valid (Use PWMOPA function: The output value at the time of cutoff is ‘Hi-Z’.)	
Output pins	Refer to Table 5-13	
A/D conversion trigger	Valid (PEAK_COUNT – DEADTIME_COUNT)	

Remark In extended complementary PWM mode, the timer RDe outputs a waveform by combining the counters and registers of timer RD0 and timer RD1.

An example of PWM output waveform in extended complementary PWM mode is shown in Figure 4-4.

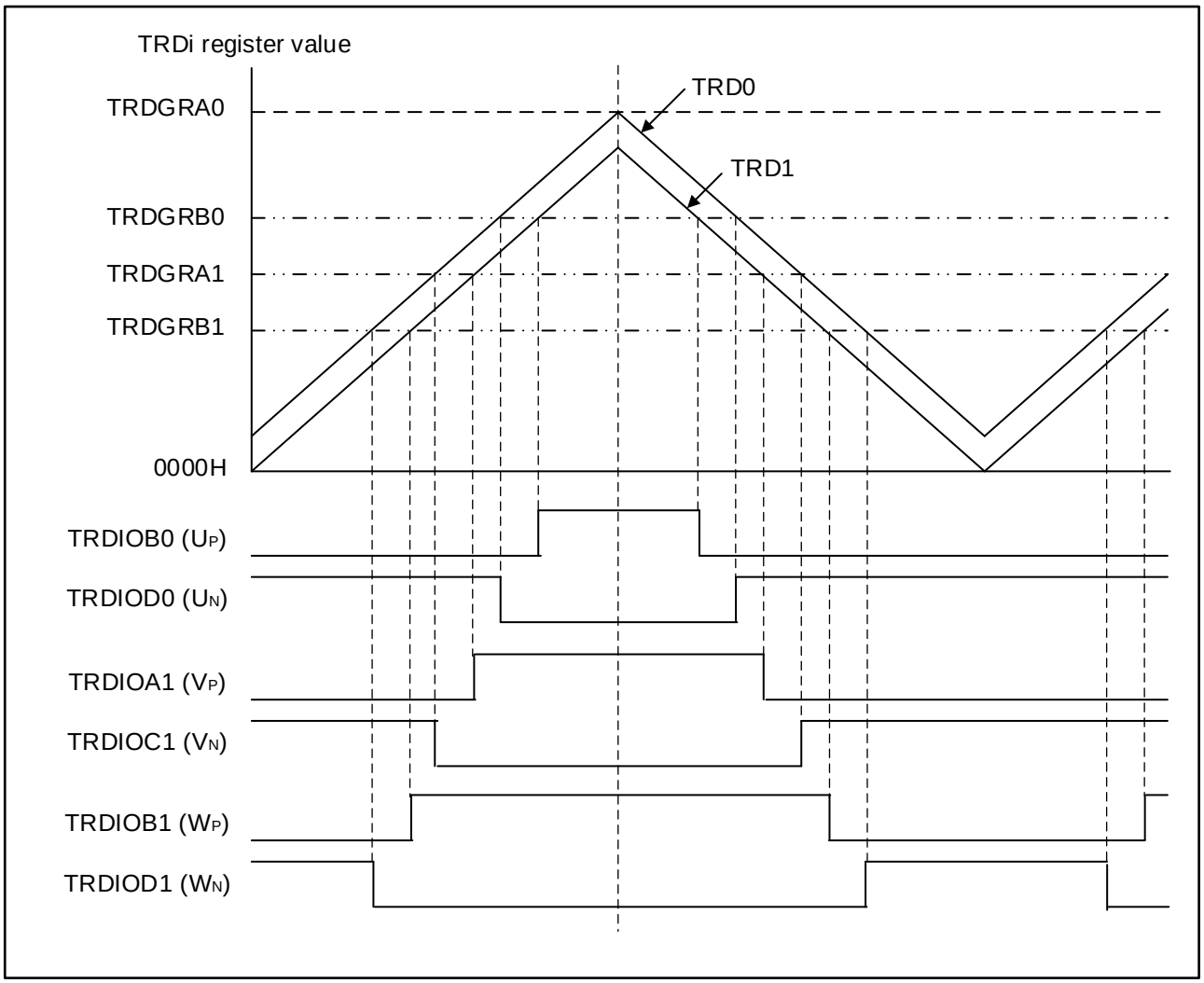

Figure 4-4 Example of PWM Output Waveform in Extended Complementary PWM Mode

4.3.1 Calculation of PWM Duty Setting Using Modulation Factor

This part summarizes how to set duty in extended complementary PWM mode (symmetric output).

Calculates the positive phase active level width from the normalized PWM duty command value and the PWM cycle count value. Set the calculated value in the buffer registers TRDGRD0, TRDGRC1 and TRDGRD1.

Positive phase active level width = PWM duty command value × PWM cycle count value TRDGRD0, TRDGRC1, TRDGRD1 = PWM cycle count value – Positive phase active level width

5. Description of the Control Software

This chapter describes the control software of this system.

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.

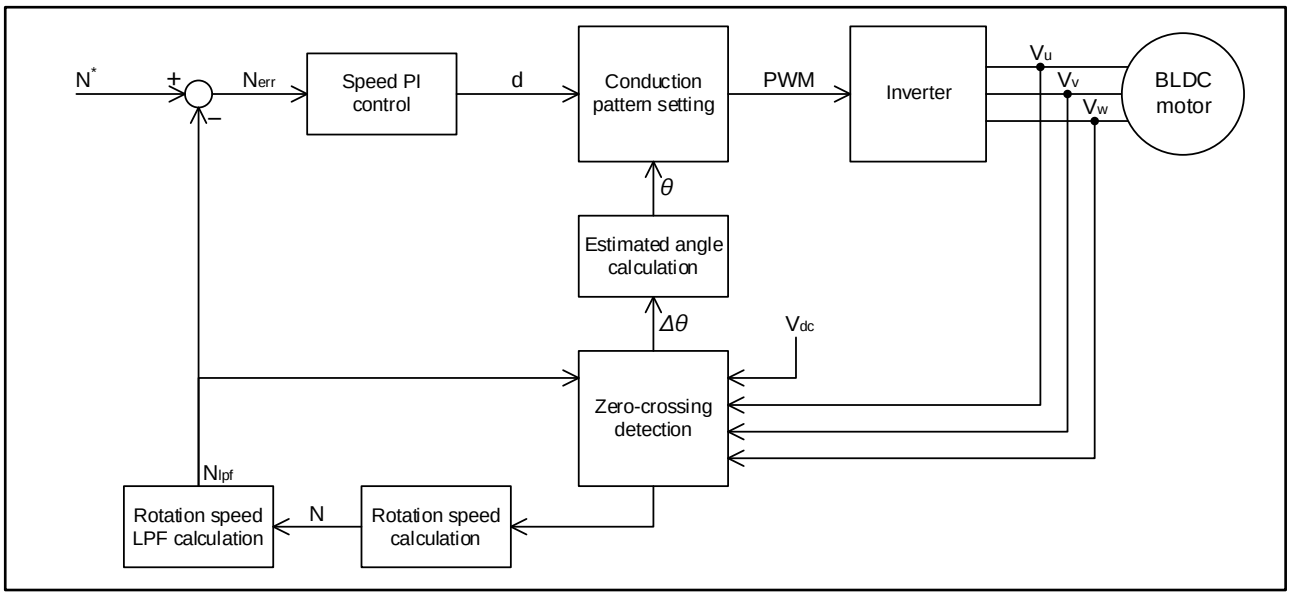

Figure 5-1 Control Block Diagram of Sample Program

Table 5-1 Variables of Control Block Diagram

Item	Variable Name	Fixed-point number format	Meaning
N	–	–	Rotation speed
N_{lpf}	g_mtr_rpm_speed	–	Rotation speed after LPF
N*	g_mtr_rpm_ref_request	–	Rotation speed command value
N_{err}	–	–	Rotation speed deviation
d	g_mtr_next_duty	–	PWM duty command value
PWM	–	–	PWM output signal
θ	g_mtr_current_angle	Q14	Estimated rotor angle position
$\Delta\theta$	g_mtr_increase_angle	Q14	Increased degree value per carrier pulse
V_{dc}	–	–	DC-bus voltage
V_U, V_V, V_W	–	–	Phase voltage

The functions of this control program are as follows.

(1) Speed PI control

Speed PI control is performed from the deviation between the rotation speed command value and the current rotation speed, and the PWM duty command value is calculated.

(2) trapezoidal control pattern setting

It switches the conducting pattern every 60-degree using the current estimated angle value, and updates the PWM output value based on the PWM duty command value. The PWM duty command value to be used is rounded to the limit value using the upper limit parameter.

(3) Inverter control

The PWM signal output from the microcontroller converts direct current into three-phase alternating current and supplies it to the BLDC motor.

(4) BLDC motor control

Converts the power supplied from the inverter into rotational torque.

(5) Zero-crossing detection

It monitors the phase voltage (V_U, V_V, V_W) of the BLDC motor and detects the zero-crossing timing of BEMF [Note]. It also corrects the estimated angle value when zero-crossing is detected and calculates the increased angle value for each carrier frequency.

(6) Estimated rotor angle position calculation

The current estimated angle value is updated by adding the increased angle value each time a PWM carrier frequency interrupt occurs.

(7) Rotation speed calculation

Calculate the current rotation speed by calculating the time elapsed in the 6 pattern switching processes from the global pulse number. The calculated current rotation speed value is used after being smoothed by the rotation speed calculation by LPF.

(8) Rotation speed calculation by LPF

The rotation speed is calculated by the exponential moving average.

Note Back electromotive force

5.2 Contents of Control

5.2.1 Motor Start / Stop / Shifting

5.2.1.1 Motor Start due to Elapsed Time after Reset

Start and stop control of the motor rotation uses elapsed time since the H/W reset is released. The motor control state and the rotational speed command value are switched by the speed control in the 1 [ms] interval timer interrupt processing.

In this sample software, the speed command value of 500 [rpm] is set after the reset release 3 [s] has elapsed, the operation mode is changed, and the motor rotation control is started.

After that, the value specified by USER_CFG_MOTOR_DIFF_SPEED (initial value: 500 [rpm]) is added to the rotation speed command value every 10 [s]. After the rotation speed command value reaches the value specified by USER_CFG_MOTOR_MAX_SPEED (initial value: 3000 [rpm]), the value specified by USER_CFG_MOTOR_DIFF_SPEED is subtracted from the rotation speed command value every 10 [s].

When the rotation speed command value becomes 0 [rpm], the operation mode is changed and the rotation control is stopped. After 3 [s] after stopping, repeat the above operation again.

5.2.1.2 Motor Start by PWM Command Input

The start and stop of rotation of the motor is controlled by inputting a PWM signal to TI05 pin. The input frequency should be between 10 [Hz] and 1 [kHz], and the operation when a signal outside the range is input is not the expected operation. In this sample software, the rotation speed command value is updated according to the duty ratio of the signal input to TI05. The relationship between the input PWM duty ratio and the rotation speed command value is as follows.

Rotation speed command value =
(PWM_COM_DUTY_MAX – “Input signal duty ratio”) × USER_CFG_MOTOR_MAX_SPEED
PWM_COM_DUTY_MAX：4096 [rpm],
USER_CFG_MOTOR_MAX_SPEED：3000 [rpm]

5.2.2 Inverter DC-bus Voltage

The inverter DC-bus voltage is measured as shown in Table 5-2. The measured voltage value is used to detect overvoltage. If an overvoltage is detected, the PWM output will stop.

Table 5-2 Inverter DC-bus Voltage Conversion Ratio

Item	Conversion Ratio (DC-bus voltage : A/D value)	A/D Input Pin
Inverter DC-bus voltage	0 [V] to 65 [V] : 000H to FFFH	ANI5

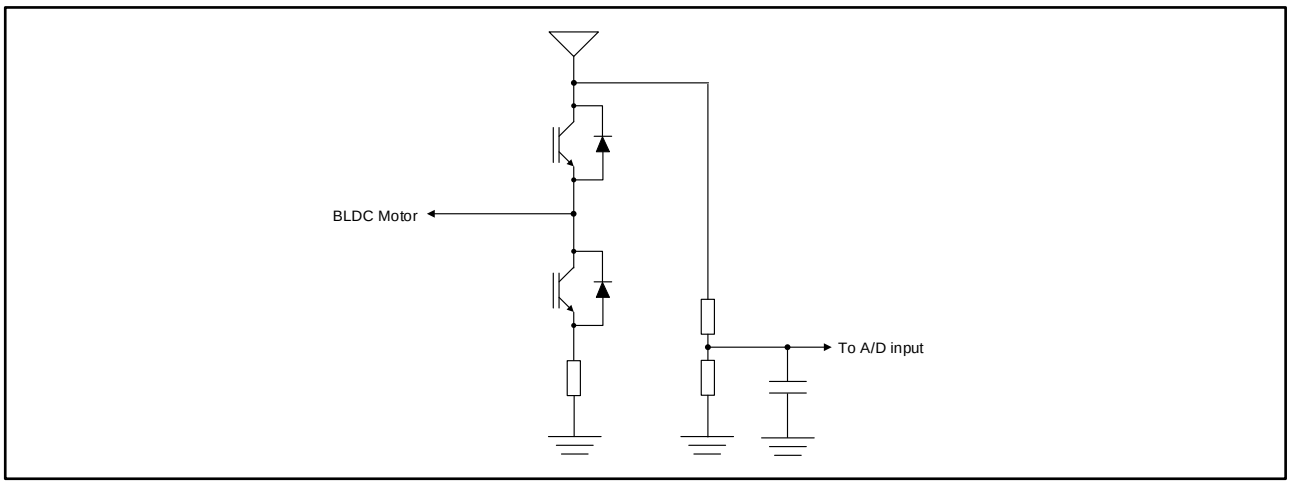

Figure 5-2 Conceptional Diagram of Inverter DC-bus Voltage Measurement Circuit

5.2.3 3-phase Voltage of Motor

Voltage of U-, V- and W-phase voltage is measured as shown in Table 5-3. The measured voltage value is used to zero-crossing detection.

Table 5-3 U, V, W Phase Voltage Conversion Ratio

Item	Conversion Ratio (3-phase voltage : A/D value)	A/D Input Pin
U, V, W phase voltage	0 [V] to 25 [V] : 000H to FFFH	ANI8, ANI9, ANI10

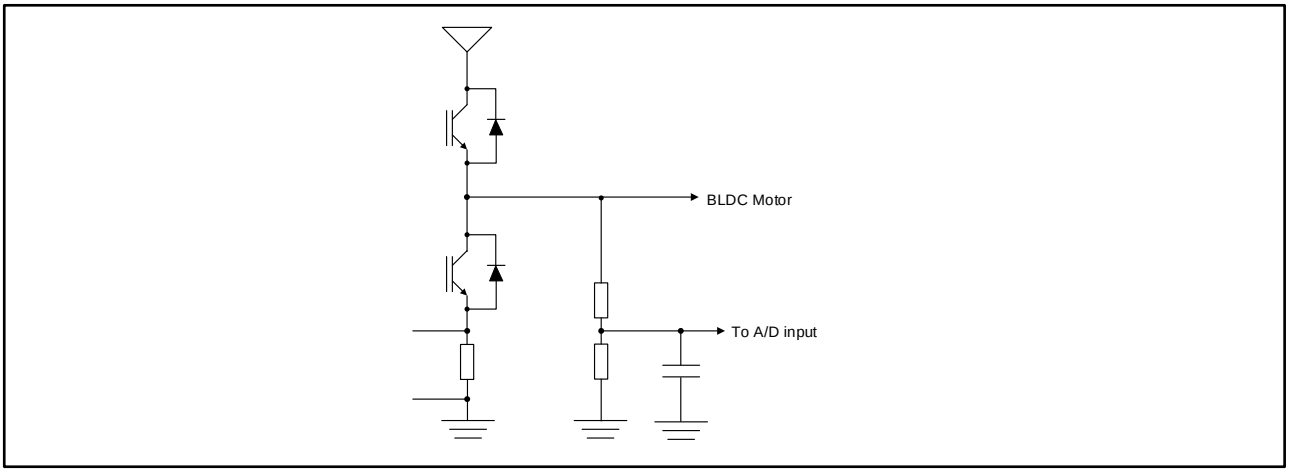

Figure 5-3 Conceptional Diagram of U-, V-, W-Phase Voltage Measurement Circuit

5.2.4 Rotation Speed Operations

The rotation speed is calculated by using zero-crossing detection and global pulse number (add for each PWM carrier cycle). The global pulse number is acquired at the timing of pattern switching when zero-crossing is detected, and the following speed calculation processing is performed using that value.

Rotation speed (N) = (60 × 20 [kHz]) / (Global pulse number × Motor pole logarithm)

Remark 20 [kHz]: PWM carrier interruption frequency

In this system, LPF process is performed on the calculation result of the rotation speed.

5.2.5 Speed PI Control

In this sample program, speed PI control is executed every 10 [ms] cycle to prevent PI control from being executed multiple times before the next conducting pattern switch.

The voltage command value (V*) is created as given below.

Proportional term (P): K_P × (Current rotation speed deviation – Last rotation speed deviation)
Integral term (I): K_I × (Current rotation speed deviation)
Voltage command value (V*) = Previous voltage command value + (P) + (I)

Remarks K_P: Proportional gain (1.50)
K_I : Integral gain (0.30)
Values of K_P and K_I depend on the used system.

For details of PI control, refer to specialized books.

5.2.6 Motor Current

Motor current is measured as shown in Figure 5-4. The value calculated by the conversion ratio is used to detect the motor current error. If a motor current error is detected, the PWM output will stop.

Table 5-4 Motor Current Voltage Conversion Ratio

Item	Conversion Ratio (Motor current (I_{dc}): A/D value)	A/D Input Pin
Motor Current	0 [A] to 50 [A]: 000H to FFFH	ANI11

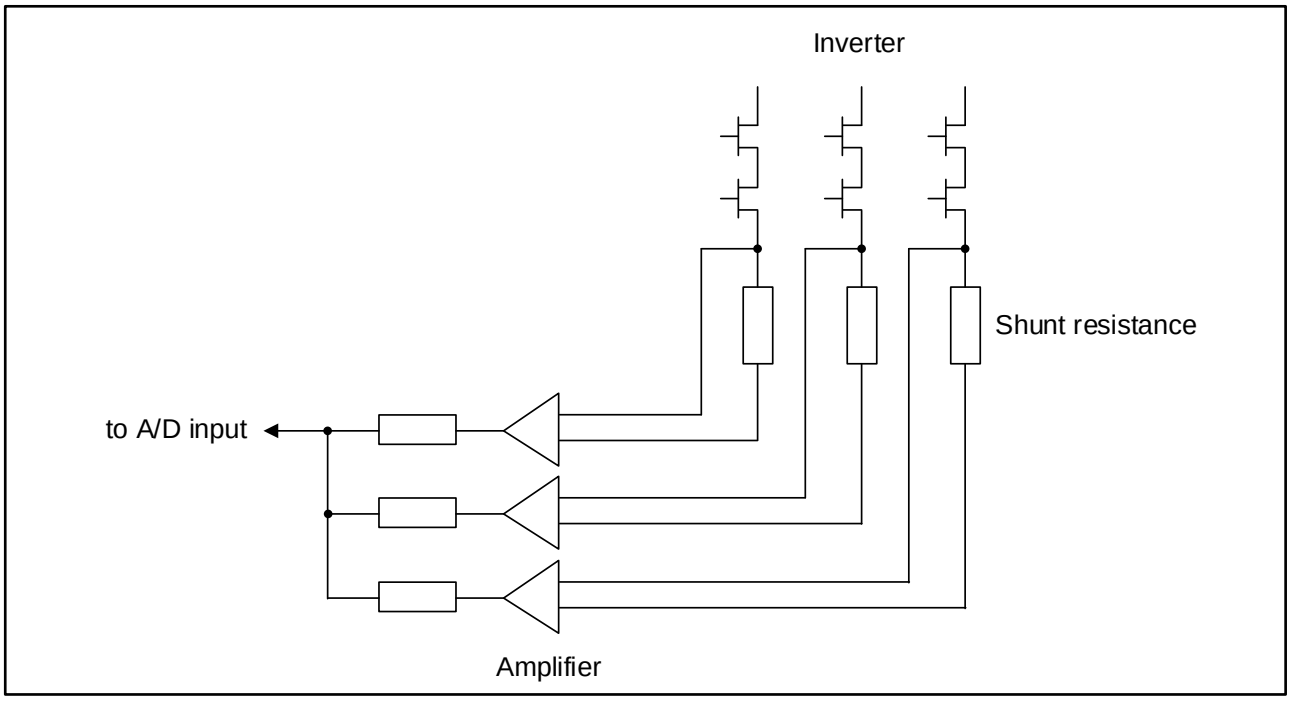

Figure 5-4 Conceptional Diagram of Motor Current Measurement Circuit

5.2.7 Motor Starting Method

Figure 5-5 shows the state transition diagram of the sample program.

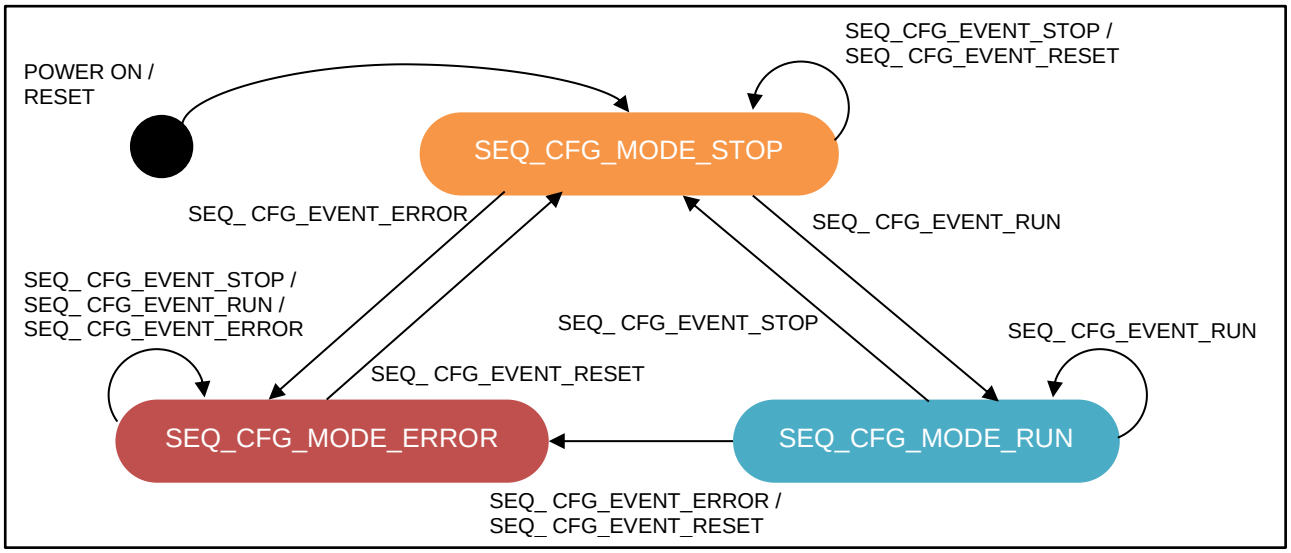

Figure 5-5 State Transition Diagram

5.2.8 System Protection Functions

This control program has the following 6 types of error status and enables emergency stop functions in case of occurrence of respective error.

(1) Overcurrent error in hardware

The pulse output is forcibly cutoff by the emergency stop signal due to overcurrent detection. Overcurrent is detected by using the internal comparator on the microcontroller (RL78/F24).

(2) Motor rotation speed error

The rotation speed calculation value is monitored at 1 [ms] intervals, and when the rotation speed exceeds the error detection threshold, the motor is stopped by the control software.

Table 5-5 Rotation Speed Error Detection Parameter

Parameter Name	Initial Value	Contents
g_err_over_speed_rpm	10000	Rotation speed error detection threshold [rpm]

(3) Motor lock error

If conducting pattern switching due to zero-crossing detection does not occur for a certain period, the motor is stopped by the control software.

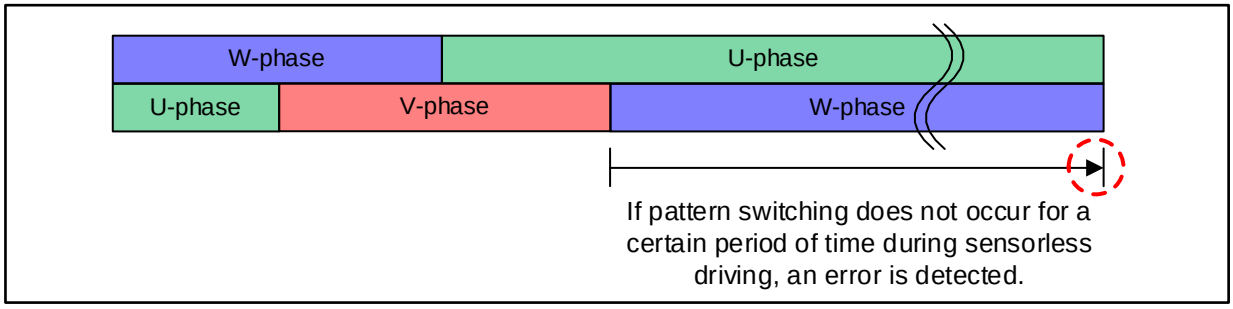

Figure 5-6 Motor Lock Error Detection

Table 5-6 Motor Lock Error Detection Parameter

Parameter Name	Initial Value	Contents
g_err_lock_detect_time	200	Motor lock error detection time [ms]

(4) DC-bus voltage error (Overvoltage and Undervoltage)

The A/D converter is used to monitor the DC-bus voltage and the motor is stopped by software when an overvoltage or undervoltage is detected.

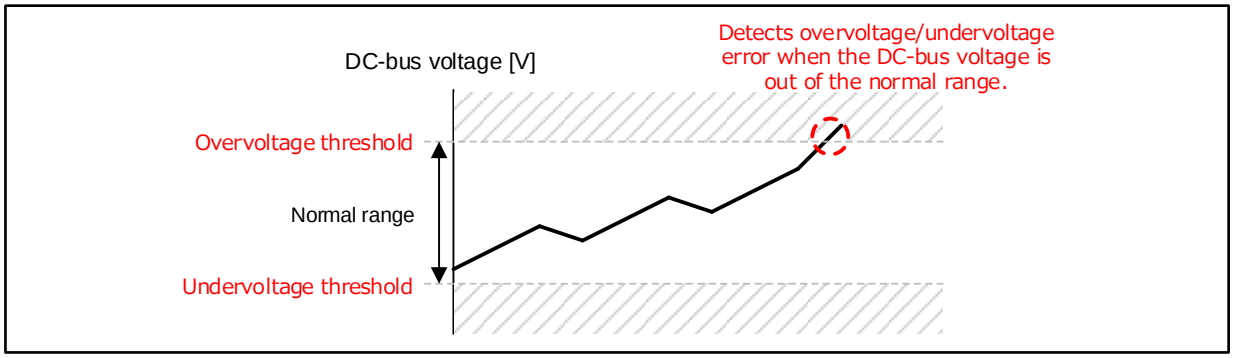

Figure 5-7 DC-bus Voltage Error Detection

Table 5-7 DC-bus Voltage Error Detection Parameters

Parameter Name	Initial Value	Fixed-point number format	Contents
g_err_over_vol_level	28.0	Q8	Overvoltage detection threshold [V]
g_err_under_vol_level	8.0	Q8	Undervoltage detection threshold [V]

(5) Motor current error

The A/D converter is used to monitor the motor current (I_{dc}) and the motor is stopped by software when an overcurrent is detected.

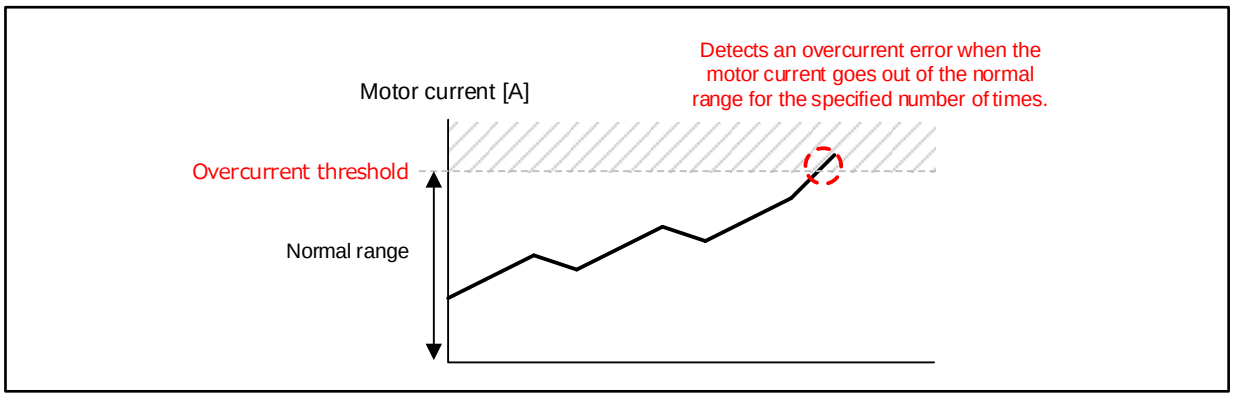

Figure 5-8 Motor Current Error Detection

Table 5-8 Motor Current Error Detection Parameters

Parameter Name	Initial Value	Fixed-point number format	Contents
g_err_over_cur_level	10.0	Q10	Overcurrent detection threshold [A]
g_err_over_cur_detect_time	3	–	Overcurrent detection time [Number of carrier cycle]

(6) Overtemperature

The A/D converter is used to monitor motor coil-end temperature and inverter board temperature and the motor is stopped by software when an overtemperature is detected.

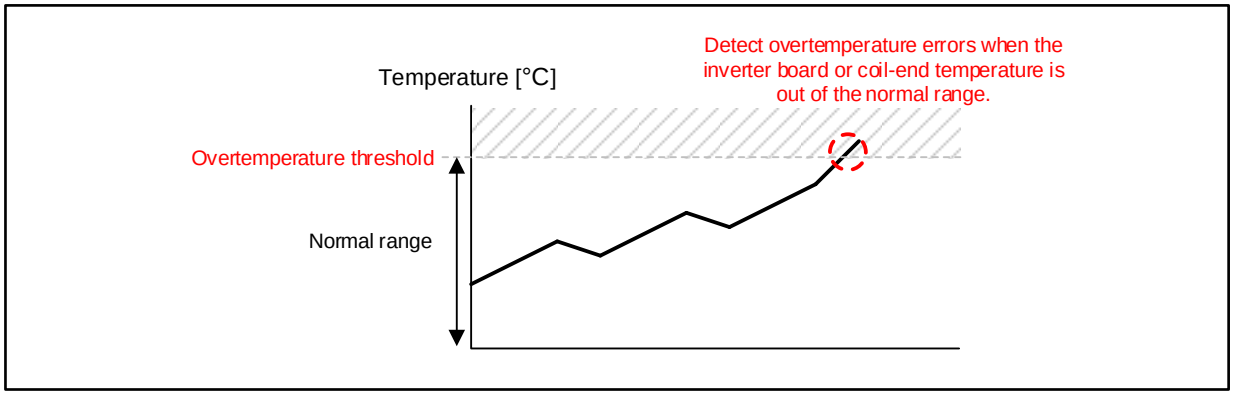

Figure 5-9 Overtemperature Detection

Table 5-9 Overtemperature Detection Parameters

Parameter Name	Initial Value	Contents
g_err_over_temp_level_board	125	Inverter board overtemperature detection threshold [°C]
g_err_over_temp_level_motor	180	Motor coil-end overtemperature detection threshold [°C]

Error status number:

If an error is detected, the number indicating the error that occurred is stored in ‘g_err_flag_cur_loop’. If multiple errors are detected, the corresponding bit will be 1.

Table 5-10 Error Status Number

Variable Name	Value	Contents
g_err_flag_cur_loop	0x0001 (bit 0)	Overvoltage error
	0x0002 (bit 1)	Undervoltage error
	0x0010 (bit 4)	Motor current error
	0x0020 (bit 5)	Overcurrent error in hardware
	0x0100 (bit 8)	Motor lock error
	0x0200 (bit 9)	Motor rotation speed error
	0x1000 (bit 12)	Inverter board overtemperature error
	0x2000 (bit 13)	Coil-end overtemperature error

5.3 System Resources

5.3.1 Interrupt Function

Table 5-11 shows the interruption resources used by this control program.

Table 5-11 Interruption Resources

Interruption	Interruption Handler	Occurrence Conditions	Main Functions
Carrier Frequency interruption (INTTRD1)	static void carrier_isr(void)	50.0 [µs] (20 [kHz])	• Zero-crossing detection processing • Conducting pattern switching • Rotation speed operation • Error monitoring
Interval timer interruption (INTTM01)	static void interval_timer_isr(void)	1 [ms] (1 [kHz))	• User interface processing • For timeout detection • Open-loop control • Minimum rotation control • Speed PI control • Get the temperature of inverter board and coil-end • Overtemperature monitoring
A/D conversion interruption (INTAD)	static void ad_completion_isr(void)	End of A/D conversion	Get DC-bus voltage, BEMF voltage, and Motor current
Capture timer interruption (INTTM05)	static void pwm_com_isr(void)	TI05 Input signal edge detection	Control by PWM signal input
One-count timer interruption (INTTM06)	static void pwm_com_timeout_isr(void)	100 [ms] (10 [Hz])	Control by PWM signal input

5.3.2 Port Function

Table 5-12 shows the port functions used in this control program.

Table 5-12 Port Function to Use

I/O	Port Number	Function	Remarks
Output	P140	Gate drivers enable/disable signal output	–
	P11	U-phase upper arm motor control signal port output (U_P)	Logic setting is 'High' active
	P12	U-phase lower arm motor control signal port output (U_N)	
	P15	V-phase upper arm motor control signal port output (V_P)	
	P16	V-phase lower arm motor control signal port output (V_N)	
	P17	W-phase upper arm motor control signal port output (W_P)	
	P30	W-phase lower arm motor control signal port output (W_N)	

5.3.3 PWM Output Function

Table 5-13 shows the PWM output function used in this control program.

Table 5-13 PWM Output Function to Use

I/O	Output Pin	Function	Remarks
Output	TRDIOB0 / P11	U-phase upper arm motor control signal output (U_P)	Logic setting is 'High' active
	TRDIOD0 / P12	U-phase lower arm motor control signal output (U_N)	
	TRDIOA1 / P15	V-phase upper arm motor control signal output (V_P)	
	TRDIOC1 / P16	V-phase lower arm motor control signal output (V_N)	
	TRDIOB1 / P17	W-phase upper arm motor control signal output (W_P)	
	TRDIOD1 / P30	W-phase lower arm motor control signal output (W_N)	

5.4 Function Specifications of the Sample Program

Multiple control functions are used in this control program. Table 5-14 shows the control functions list used in this control program.

For detailed processing, refer to flowcharts or source files.

Table 5-14 List of Control Functions (1/5)

Function Name	Processing Overview
File Name: mtr_main.c	
Name: main () Input: None Output: None	• Call hardware initialization function – Call the CPU initialization function – Call the inverter dependency initialization function – Call the motor dependency initialization function – Call the user initialization function • Main processing – Call the watchdog timer clear function
File Name: mtr_cpu_setting.c (1/2)	
Name: cpu_init () Input: None Output None	CPU initialization function • Initialization function call of each peripheral function – Call the clock initialization function – Call the I/O port initialization function – Call the TAU01 initialization function – Call the A/D converter initialization function – Call the D/A converter initialization function – Call the comparator initialization function – Call the Timer RDe initialization function – Call the TAU05 initialization function – Call the TAU06 initialization function • Timer count start function call of each timer – Call the TAU01 count start function – Call the Timer RDe count start function – Call the TAU05 count start function – Call the TAU06 count start function
Name: cgc_init () Input: None Output: None	Clock initialization function
Name: port_init () Input: None Output: None	I/O port initialization function
Name: tau01_init () Input: uint16_t freq Output: None	TAU01 initialization function freq: TAU01 interrupt period value
Name: tau01_start () Input: None Output: None	TAU01 count start function
Name: trde_init () Input: None Output: None	Timer RDe initialization function
Name: trde_start () Input: None Output: None	Timer RDe count start function

Table 5-14 List of Control Functions (2/5)

Function Name	Processing Overview
File Name: mtr_cpu_setting.c (2/2)	
Name: trde_disable_inv_out () Input: None Output: None	Inverter output stop function
Name: pwmopa_release_hz () Input: None Output: None	Forcibly cutoff release function for PWM output
Name: adc_init () Input: None Output: None	A/D converter initialization function
Name: adc_set_ad_ch_bemf () Input: uint16_t channel Output: None	Function to get the A/D conversion result channel: A/D channel number
Name: dac_init () Input: None Output: None	D/A converter initialization function
Name: cmp_init () Input: None Output: None	Comparator initialization function
Name: port_change_pattern () Input: uint8_t pattern Output: None	Conducting pattern update function pattern: Conducting pattern
Neme: trde_calc_pwm_ref () Input: int16_t ref Output: None	Calculates the value to be set in the timer RDe PWM duty setting register from the set duty value. ref: PWM duty command value
Name: tau05_init () Input: None Output: None	TAU05 initialization function
Name: tau05_start () Input: None Output: None	TAU05 count start function
Name: tau06_init () Input: None Output: None	TAU06 initialization function
Name: tau06_start () Input: None Output: None	TAU06 count start function
Name: wdt_clear () Input: None Output: None	Watchdog timer counter clear function

Table 5-14 List of Control Functions (3/5)

Function Name	Processing Overview
File Name: mtr_fix_calc.c	
Name: fix8_mul_int16 () Input: int16_t fix_1, fix_2 Output: int16_t temp	Fixed-point calculation function: (16-bit × 16-bit) >> 8 fix_1: Input data 1 fix_2: Input data 2 temp: Calculation result
Name: fix10_mul_int16 () Input: int16_t fix_1, fix_2 Output: int16_t temp	Fixed-point calculation function: (16 bit × 16 bit) >> 10 fix_1: Input data 1 fix_2: Input data 2 temp: Calculation result
Name: fix12_mul_int16 () Input: int16_t fix_1, fix_2 Output: int16_t temp	Fixed-point calculation function: (16 bit × 16 bit) >> 12 fix_1: Input data 1 fix_2: Input data 2 temp: Calculation result
Name: fix14_mul_int16 () Input: int16_t fix_1, fix_2 Output: int16_t temp	Fixed-point calculation function: (16 bit × 16 bit) >> 14 fix_1: Input data 1 fix_2: Input data 2 temp: Calculation result
Name: fix16_mul_int16 () Input: int16_t fix_1, fix_2 Output: int16_t temp	Fixed-point calculation function: (16 bit × 16 bit) >> 16 fix_1: Input data 1 fix_2: Input data 2 temp: Calculation result
File Name: mtr_function.c	
Name: err_check_error_cur_loop () Input: None Output: None	Error checking function that occurred • Overcurrent detection • Overvoltage/undervoltage detection • Call the motor lock error detection function • Hardware overcurrent detection • Motor rotation speed error detection
Name: err_check_over_temp () Input: None Output: None	Overtemperature error detection • Inverter board overtemperature detection • Motor coil-end overtemperature detection
Name: err_error_stop () Input: None Output: None	Motor stop function due to error detection • Call the inverter output stop function
Name: mtr_detect_synchro_loss () Input: None Output: uint8_t detect	Motor lock error detection function detect: Error result
File Name: mtr_interrupt_bemf.c (1/2)	
Name: mtr_detect_zerocross () Input: None Output: None	Zero-crossing detection function
Name: mtr_speed_calc () Input: None Output: None	Motor rotation speed calculation function
Name: pwm_com_get_duty () Input: None Output: uint32_t s_pwm_com_duty	Function to get the duty ratio of the input PWM signal s_pwm_com_duty: Duty ratio of the input PWM signal

Table 5-14 List of Control Functions (4/5)

Function Name	Processing Overview
File Name: mtr_interrupt_bemf.c (2/2)	
Name: interval_timer_isr () Input: None Output: None	1 [ms] interval timer interrupt processing • User interface processing • Generate motor lock error detection timing • Open-loop control • Low speed rotation control • Speed PI control • Sequence processing • Overtemperature detection
Name: carrier_isr () Input: None Output: None	Carrier frequency synchronous interrupt processing • Conducting pattern switching • Call the motor rotation speed calculation function • Non-energizing phase A/D conversion channel setting • DC-bus voltage calculation • Call the zero-crossing detection function • Call the duty value setting function • Call the error checking function
Name: ad_completion_isr () Input: None Output: None	End of A/D conversion interrupt processing • Get the A/D value of DC-bus voltage • Get the A/D value of non-energizing phase voltage • Get the A/D value of motor current • Motor current calculation
Name: pwm_com_isr () Input: None Output: None	Function to measure the pulse width of the input PWM signal
Name: pwm_com_timeout_isr () Input: None Output: None	Time-out detection function for pulse width measurement of the input PWM signal
File Name: mtr_inv_setting.c	
Name: inv_init () Input: None Output: None	Inverter dependency initialization function • DC-bus voltage physical value conversion coefficient setting • Motor current physical value conversion coefficient setting
Name: th_get_temp_board () Input: None Output: float32_t temp1	Get the inverter board temperature temp1: Inverter board temperature
Name: th_get_temp_motor () Input: None Output: float32_t temp1	Get the coil-end temperature temp1: Coil-end temperature

Table 5-14 List of Control Functions (5/5)

Function Name	Processing Overview
File Name: r_mtr_sequence_api.c	
Name: R_SEQ_ExecEvent () Input: uint8_t event Output: None	• Status update • Function call of the event that occurred event: Event that occurred
Name: seq_act_run () Input: uint8_t state Output: uint8_t state	Call the variable initialization function when driving the motor state: Sequence status data
Name: seq_act_stop () Input: uint8_t state Output: uint8_t state	• Call the inverter stop function • Call the variable initialization function when driving the motor • Call the PWM output forcibly cutoff release function state: Sequence status data
Name: seq_act_none () Input: uint8_t state Output: uint8_t state	No operation state: Sequence status data
Name: seq_act_reset () Input: uint8_t state Output: uint8_t state	• Call the variable initialization function when driving the motor • Call the PWM output forcibly cutoff release function state: Sequence status data
Name: seq_act_error () Input: uint8_t state Output: uint8_t state	Call the inverter output stop function state: Sequence status data
Name: seq_act_break () Input: uint8_t state Output: uint8_t state	• Break count setting • Call the conducting pattern switching function state: Sequence status data
Name: seq_init_start () Input: None Output: None	Variable initialization function when driving the motor
File Name: r_mtr_user_control_api.c	
Name: R_USER_InitMotor () Input: None Output: None	Motor dependency initialization function • Motor parameters initialization
Name: R_USER_InitCtrl () Input: None Output: None	User parameters initialization function • User setting parameters initialization
Name: R_USER_MotorControl () Input: None Output: None	User interface processing • Update motor status (Start driving the motor) • Rotation speed command value setting
Name: user_change_ref_speed () Input: None Output: None	Update of rotation speed command value

5.5 Variable Specifications of the Sample Program

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

Caution: Do not change variables with (*) in this sample program.

Table 5-15 Variables List (1/3)

Variables	Type	Content
File Name: mtr_function.c		
g_err_over_vol_level	int16_t	Overvoltage detection threshold
g_err_under_vol_level	int16_t	Undervoltage detection threshold
g_err_over_cur_level	int16_t	Overcurrent detection threshold
g_err_over_cur_detect_time	uint16_t	Overcurrent detection time
g_err_over_cur_time_cnt	uint16_t	Overcurrent detection time count value (*)
g_err_lock_detect_time	uint16_t	Motor lock error detection time
g_err_flag_cur_loop	uint16_t	Error detection flag (*)
g_err_timeout_cnt	uint16_t	Motor lock error detection time count value (*)
g_err_over_speed_rpm	uint16_t	Motor rotation speed error detection threshold
g_err_over_temp_level_board	uint16_t	Inverter board overtemperature detection threshold
g_err_over_temp_level_motor	uint16_t	Coil-end overtemperature detection threshold
File Name: mtr_inv_setting.c		
g_inv_vdc_range	int16_t	DC-bus voltage conversion coefficient (*)
g_inv_cur_range	int16_t	Motor current conversion coefficient (*)
g_inv_bemf_range	int16_t	BEMF voltage conversion coefficient (*)
File Name: r_mtr_user_control_api.c		
g_user_TargetRotateDir	uint8_t	Rotation direction [0: CW, 1: CCW]
g_user_MotorAccelDir	uint8_t	Acceleration / deceleration state of motor (*)
File Name: mtr_interrupt_bemf.c (1/3)		
g_mtr_rotate_dir	uint8_t	Current motor rotation direction (*) [0: CW, 1: CCW]
g_mtr_mode_system_request	uint8_t	User event input [0: Stop, 1: Start, 2: Error, 3: Reset. 4: Break]
g_mtr_mode_system	uint8_t	Operating condition (*) [0: Stopping, 1: Driving, 2: Error is occurring]
g_err_error_status	uint8_t	Error identification number (*)
g_mtr_run_mode	uint8_t	Driving mode (*) [0: Open-loop, 1: BEMF]
g_mtr_rpm_speed	int16_t	Current rotation speed (*)
g_mtr_ref_limit	int16_t	PWM duty limit
g_mtr_rpm_ref	int16_t	Target rotation speed (*)
g_mtr_rpm_ref_request	int16_t	Target rotation speed command value
g_mtr_rpm_ref_request_old	int16_t	Previous target rotation speed command value (*)
g_mtr_slope_time_count	uint8_t	Target rotation speed slope control counter (*)
g_mtr_setting_min_rpm	uint16_t	Minimum rotation speed setting value
g_mtr_speed_lpf_factor	uint16_t	Rotation speed filter setting value
g_mtr_speed_culc_counter	uint8_t	Number of segments for rotation speed calculation (*)

Table 5-15 Variables List (2/3)

Variables	Type	Content
File Name: mtr_interrupt_bemf.c (2/3)		
g_ad_data_bemf	uint16_t	A/D value of non-energizing phase voltage (*)
g_ad_data_vdc	uint16_t	A/D value of DC-bus voltage (*)
g_ad_data_idc	int16_t	A/D value of motor current (*)
g_ad_data_temp_board	uint16_t	A/D value of inverter board temperature (*)
g_ad_data_temp_motor	uint16_t	A/D value of coil-end temperature (*)
g_mtr_time_setting_offset	uint16_t	Offset initialization time setting when driving the motor
g_mtr_time_cnt_offset	uint16_t	Offset initialization counter when driving the motor (*)
g_mtr_vdc_value	int16_t	Physical value of DC-bus voltage
g_mtr_idc_value	int16_t	Physical value of motor current
g_mtr_offset_idc	int16_t	Motor current offset value
g_mtr_offset_idc_lpf_factor	int16_t	Motor current offset filter value
g_mtr_idc_lpf_factor	uint16_t	Motor current filter value
g_mtr_vdc_lpf_factor	uint16_t	DC-bus voltage filter value
g_th_cur_temp_board	float32_t	Physical value of inverter board temperature
g_th_cur_temp_motor	float32_t	Physical value of coil-end temperature
g_mtr_current_angle	uint16_t	Estimated angle value (*)
g_mtr_increase_angle	uint16_t	Incremental angle value for each carrier frequency (*)
g_mtr_const_rpm_to_inc_angle	int16_t	Conversion coefficient from rotation speed (RPM) to angle value
g_mtr_speed_pi_cnt	uint8_t	Speed PI control timing generation counter (*)
g_mtr_speed_kp_factor	int16_t	Term (P) coefficient for speed PI control
g_mtr_speed_ki_factor	int16_t	Term (I) coefficient for speed PI control
g_mtr_speed_ref	int16_t	PWM duty command value for speed PI control (*)
g_mtr_speed_ref_p	int16_t	Term (P) calculation value for speed PI control (*)
g_mtr_speed_ref_i	int16_t	Term (I) calculation value for speed PI control (*)
g_mtr_speed_delta_pre	int16_t	Time-lag deviation of first order for speed PI control (*)
g_mtr_speed_pi_err_limit	int16_t	Deviation limit value for speed PI control
g_mtr_speed_pi_interval	uint8_t	Speed PI control interval
g_mtr_pole_pairs	uint8_t	Number of pole pairs for motor
g_mtr_rpm_slope	int16_t	Target rotation speed conversion rate (*)
g_mtr_rpm_slope_request	int16_t	Target rotation speed conversion rate command value
g_mtr_rpm_slope_ol	int16_t	Target rotation speed conversion rate during an open-loop operation
g_mtr_arign_cnt_ol_1st	uint16_t	1st position determination processing time counter (*)
g_mtr_setting_arign_time_ol_1st	uint16_t	1st position determination processing setting time
g_mtr_arign_cnt_ol_2nd	uint16_t	2nd position determination processing time counter (*)
g_mtr_setting_arign_time_ol_2nd	uint16_t	2nd position determination processing setting time
g_mtr_open_loop_mode	uint16_t	Operation mode during an open-loop operation [1: 1st position determination, 2: 2nd position determination, 3: Open-loop operation]
g_mtr_change_bemf_rpm_ol	uint16_t	Sensorless control switching speed
g_mtr_initial_angle_ol_1st	int16_t	Angle data of 1st position determination processing
g_mtr_initial_angle_ol_2nd	int16_t	Angle data of 2nd position determination processing
g_mtr_current_pattern	uint8_t	Current conducting pattern number (*)
g_mtr_next_pattern	uint8_t	Conducting pattern number update reservation (*)

Table 5-15 Variables List (3/3)

Variables	Type	Content
File Name: mtr_interrupt_bemf.c (3/3)		
g_mtr_float_port	uint8_t	Get the number of non-energizing phase port (*)
g_mtr_pattern_change_enable	uint8_t	Conducting pattern update permission flag (*)
g_mtr_pattern_table_120deg_cw	st_pattern_table_t	trapezoidal control pattern table [CW] (*)
g_mtr_pattern_table_120deg_ccw	st_pattern_table_t	trapezoidal control pattern table [CCW] (*)
* gp_pattern_table	st_pattern_table_t	Conducting pattern table pointer (*)
g_mtr_detect_mode	uint8_t	BEMF detection mode [0: BEMF no detection, 1: BEMF detection]
g_mtr_segment_pulse_number	uint16_t	Segment pulse number (*)
g_mtr_global_pulse_number	uint16_t	Global pulse number (*)
g_mtr_global_pulse_number_old	uint16_t	Global pulse number when the rotation speed is fixed (*)
g_mtr_permit_bemf_mode	uint8_t	BEMF mode transition permission flag (*)
g_mtr_advance_angle	uint16_t	Advance angle value (*)
g_mtr_bemf_detect_end_threshold	uint16_t	Zero-crossing detection voltage
g_mtr_correction_angle	uint16_t	Correction angle value (*)
g_mtr_detect_slope	uint8_t	BEMF detection slope value (*)
g_mtr_initial_duty	int16_t	Duty initial setting value
g_mtr_next_duty	int16_t	Duty update reservation value (*)
g_mtr_current_duty	int16_t	Current duty value (*)
g_mtr_auto_power_off_time	uint16_t	Time for the motor to stop automatically (*)
g_mtr_auto_power_off_tick_count	uint16_t	Counter for the time when the motor automatically stops (*)
g_mtr_brake_count	uint16_t	Break count (*)
g_mtr_brake_count_request	uint16_t	Break count command value (*)
g_mtr_brake_tick_count	uint16_t	Counter for the break count detection (*)

5.6 Macro Definitions of the Sample Program

Table 5-16 shows a list of macro definitions used in this control program.

Table 5-16 Macro Definitions List (1/4)

Macro Name	Value	Content
File Name: mtr_cpu_setting.h		
MCUIO_PLL_CLOCK_HZ	80000000	PLL clock frequency [Hz]
MCUIO_SYSTEM_CLOCK_HZ	40000000	CPU clock frequency [Hz]
MCUIO_PERIPHERAL_CLOCK_HZ	40000000	Peripheral hardware clock frequency [Hz]
File Name: mtr_interrupt_bemf.h		
MTR_MODE_RUN_OPENLOOP	1	Operation mode: Open-loop mode
MTR_MODE_RUN_BEMF	2	Operation mode: BEMF mode
MTR_MODE_OPENLOOP_ARIGN_1ST	1	1st position determination processing
MTR_MODE_OPENLOOP_ARIGN_2ND	2	2nd position determination processing
MTR_MODE_OPENLOOP_ROTATE	3	Open-loop driving in open-loop mode
ERROR_OVER_VOLTAGE	0x0001	Overvoltage detection
ERROR_UNDER_VOLTAGE	0x0002	Undervoltage detection
ERROR_OVER_CURRENT	0x0010	Overcurrent detection
ERROR_OVER_CURRENT_HW	0x0020	H/W overcurrent detection
ERROR_STEP_OUT	0x0100	Motor lock error detection
ERROR_OVER_SPEED	0x0200	Motor rotation speed error detection
ERROR_OVER_TEMPERATURE_BOARD	0x1000	Inverter board overtemperature detection
ERROR_OVER_TEMPERATURE_MOTOR	0x2000	Coil-end overtemperature detection
MTR_SLOPE_UP	0	BEMF detection in up slope
MTR_SLOPE_DOWN	1	BEMF detection in down slope
MTR_PATTERN_NONE	0	No pattern
MTR_U_PWM_V_LOW_W_OFF	1	Conducting pattern: U-phase to V-phase
MTR_U_PWM_W_LOW_V_OFF	2	Conducting pattern: U-phase to W-phase
MTR_V_PWM_W_LOW_U_OFF	3	Conducting pattern: V-phase to W-phase
MTR_V_PWM_U_LOW_W_OFF	4	Conducting pattern: V-phase to U-phase
MTR_W_PWM_U_LOW_V_OFF	5	Conducting pattern: W-phase to U-phase
MTR_W_PWM_V_LOW_U_OFF	6	Conducting pattern: W-phase to V-phase
MTR_PATTERN_NUM	6	Total number of conducting patterns
MTR_ALL_LOW_ON	7	Output pattern: Break (stop conducting pattern)
MTR_BEMF_MODE_NONE	0	No detection of BEMF
MTR_BEMF_MODE_DETECT	1	BEMF detection
MTR_BEMF_DETECT_GUARD_CNT	2	Number of carrier cycles for zero-crossing detection invalid period
ANGLE_30_DEGREE	1366	Normalized degree value: 30-degree
ANGLE_60_DEGREE	2731	Normalized degree value: 60-degree
ANGLE_90_DEGREE	4096	Normalized degree value: 90-degree
ANGLE_150_DEGREE	6827	Normalized degree value: 150-degree
ANGLE_210_DEGREE	9558	Normalized degree value: 210-degree
ANGLE_270_DEGREE	12288	Normalized degree value: 270-degree
ANGLE_330_DEGREE	15019	Normalized degree value: 330-degree
ANGLE_360_DEGREE	16384	Normalized degree value: 360-degree
PWM_COM_DUTY_MAX	4096	Maximum value of PWM input (duty = 100%)
PWM_COM_DUTY_MIN	0	Minimum value of PWM input (duty = 0%)

Table 5-16 Macro Definitions List (2/4)

Macro Name	Value	Content
File Name: mtr_inv_setting.h		
INV_VDC_VOL_DIV	13	DC-bus voltage resistance voltage divider coefficient
INV_IDC_CUR_RANGE	50.0f	Input range of motor current
INV_BEMF_VOL_DIV	5	BEMF voltage resistance voltage divider coefficient
File Name: r_mtr_sequence_api.h		
SEQ_CFG_MODE_STOP	0x00	Operation mode: Stop
SEQ_CFG_MODE_RUN	0x01	Operation mode: Driving
SEQ_CFG_MODE_ERROR	0x02	Operation mode: Error
SEQ_CFG_MODE_BRAKE	0x03	Operation mode: Break
SEQ_CFG_SIZE_STATE	4	Total number of operation modes
SEQ_CFG_EVENT_STOP	0x00	User event input: Stop
SEQ_CFG_EVENT_RUN	0x01	User event input: Driving
SEQ_CFG_EVENT_ERROR	0x02	User event input: Error
SEQ_CFG_EVENT_RESET	0x03	User event input: Reset
SEQ_CFG_EVENT_BRAKE	0x04	User event input: Break
SEQ_CFG_SIZE_EVENT	5	Total number of user event input
SEQ_CFG_UNKNOWN_ERROR	0xFF	User event input: No event
File Name: r_mtr_user_control_cfg.h (1/2)		
USER_CFG_MOTOR_POLE_PAIRS	2	Number of pole pairs for motor
USER_CFG_MOTOR_MAX_SPEED	3000	Rotation speed upper limit (mechanical angle) [rpm]
USER_CFG_REF_SPEED_ACCEL	0	Acceleration command status
USER_CFG_REF_SPEED_DECEL	1	Deceleration command status
USER_CFG_MOTOR_INITIAL_DIR	0	Rotation direction [0: CW, 1: CCW]
USER_CFG_MOTOR_DIFF_SPEED	500	Acceleration / deceleration difference value (mechanical angle) [rpm]
USER_CFG_TIME_SETTING_OFFSET	30000	Offset setting time at startup (number of carrier pulses)
USER_CFG_TIME_CNT_OFFSET	0	Offset counter at startup
USER_CFG_OFFSET_IDC	0	Offset initial value of motor current
USER_CFG_OFFSET_IDC_LPF_FACTOR	0.1f	Offset filter value of motor current
USER_CFG_SPEED_PI_INTERVAL	10	Speed PI control period [ms]
USER_CFG_SPEED_PI_ERROR_LIMIT	9000	Upper limit of rotation speed error by speed PI control
USER_CFG_KP_FACTOR	1.50f	Proportional gain coefficient of speed PI control (term: K_P)
USER_CFG_KI_FACTOR	0.30f	Integral gain coefficient of speed PI control (term: K_I)
USER_CFG_INITIAL_RPM_SLOPE	0	Target speed update rate initial value
USER_CFG_RPM_SLOPE	10	Target speed update rate
USER_CFG_INITIAL_RPM_REF	0.0	Initial value of target speed
USER_CFG_RPM_CLOSE_TO_OPEN	500	Open-loop rush speed (mechanical angle) [rpm]
USER_CFG_RPM_SLOPE_OL	1	Target speed update rate in open-loop mode

Table 5-16 Macro Definitions List (3/4)

Macro Name	Value	Content
File Name: r_mtr_user_control_cfg.h (2/2)		
USER_CFG_RPM_OPEN_TO_CLOSE	600	Closed-loop rush speed (mechanical angle) [rpm]
USER_CFG_ARIGN_TIME_OL_1ST	200	1st position determination processing time [ms]
USER_CFG_ARIGN_TIME_OL_2ND	20	2nd position determination processing time [ms]
USER_CFG_ARIGN_DEGREES_OL_1ST	5462	1st position determination processing angle value [degree]
USER_CFG_ARIGN_DEGREES_OL_2ND	0	2nd position determination processing angle value [degree]
USER_CFG_INITIAL_DUTY_OL	0.20f	Normalized output duty in open-loop mode
USER_CFG_DUTY_LIMIT	0.95f	Upper limit of duty
USER_CFG_IDC_LPF_FACTOR	0.10f	LPF value of Idc
USER_CFG_VDC_LPF_FACTOR	0.25f	LPF value of V_{dc}
USER_CFG_SPEED_LPF_FACTOR	0.40f	LPF value of rotation speed
USER_CFG_BEMF_DETECT_END_TH	30	Zero-crossing detection voltage threshold [A/D value]
USER_CFG_ADVANCE_ANGLE	0	Advance angle value
USER_CFG_OVER_VOLTAGE_LEVEL	28.0f	Overvoltage detection threshold [V]
USER_CFG_UNDER_VOLTAGE_LEVEL	8.0f	Undervoltage detection threshold [V]
USER_CFG_OVER_CUR_LEVEL_HW_CUR	20.0f	H/W overcurrent detection threshold [A]
USER_CFG_OVER_CUR_LEVEL_SW	10.0f	Overcurrent detection threshold [A]
USER_CFG_OVER_CUR_DETECT_TIME	3	Overcurrent detection timer [Number of carrier pulses]
USER_CFG_LOCK_DETECT_TIME	200	Motor lock error detection time [ms]
USER_CFG_OVER_SPEED_RPM	10000	Rotation speed error detection speed (mechanical angle) [rpm]
USER_CFG_OVER_CUR_LEVEL_HW	2.5f	H/W motor current error detection threshold [V]
USER_CFG_OVER_TEMP_BOARD	125	Inverter board overtemperature detection threshold [°C]
USER_CFG_OVER_TEMP_MOTOR	180	Coil-end overtemperature detection threshold [°C]
USER_CFG_AUTO_POWEROFF_TIME	0	Time until the motor stops automatically [s]
USER_CFG_BRAKE_TIME	2	Break time [s]
USER_CFG_TIME_CONTROL	–	If defined: Rotation speed is updated over time If node defined: Rotation speed is updated with the input PWM signal

Table 5-16 Macro Definitions List (4/4)

Macro Name	Value	Content
File Name: mtr_fix_calc.h		
FIX_C (x, b)	((int16_t) (((float32_t) (x)) * ((float32_t) (1UL << (b)))))	Shifts the variable of 16-bit to the left by the specified number of bits
LFIX_C (x, b)	((int32_t) (((float32_t) (x)) * ((float32_t) (1UL << (b)))))	Shifts the variable of 32-bit to the left by the specified number of bits

5.7 Flowchart of the Sample Program

5.7.1 Main Processing Function

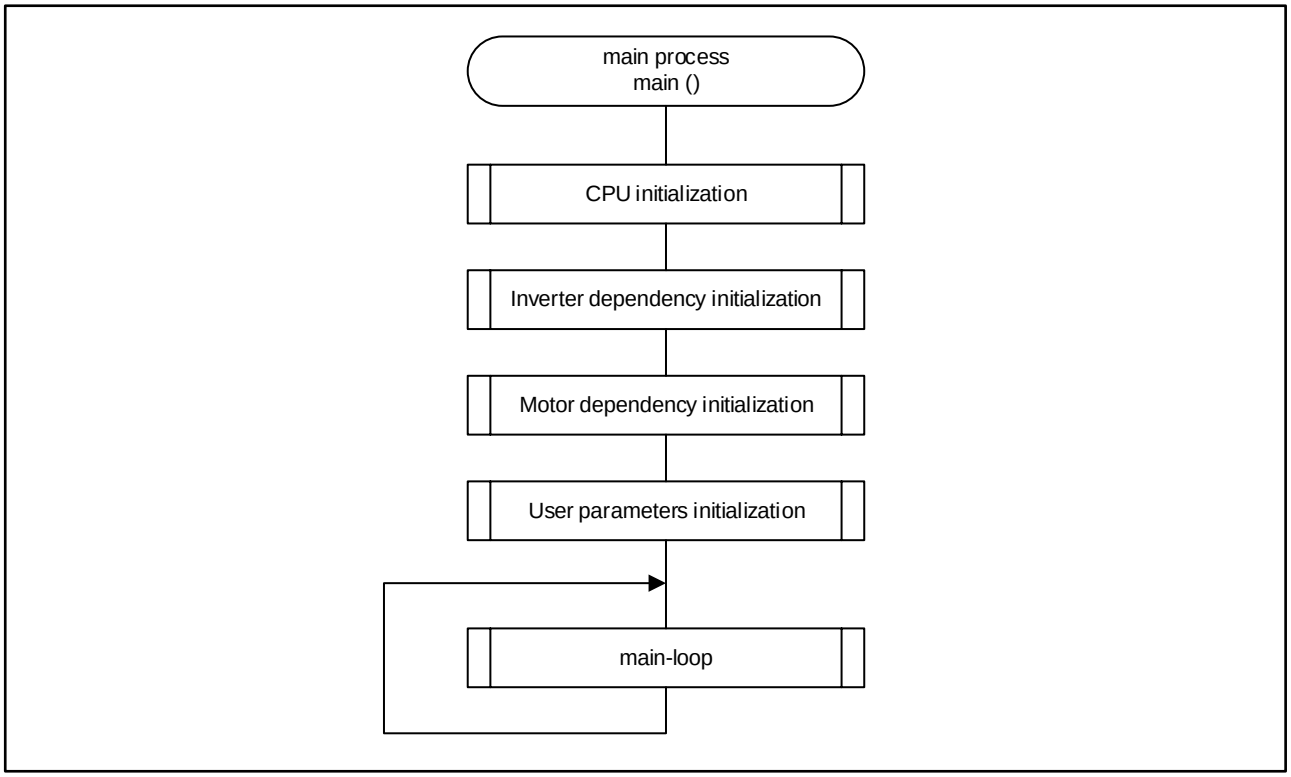

Figure 5-10 Main Processing Function

5.7.2 1 [ms] Interval Timer Interrupt Handler

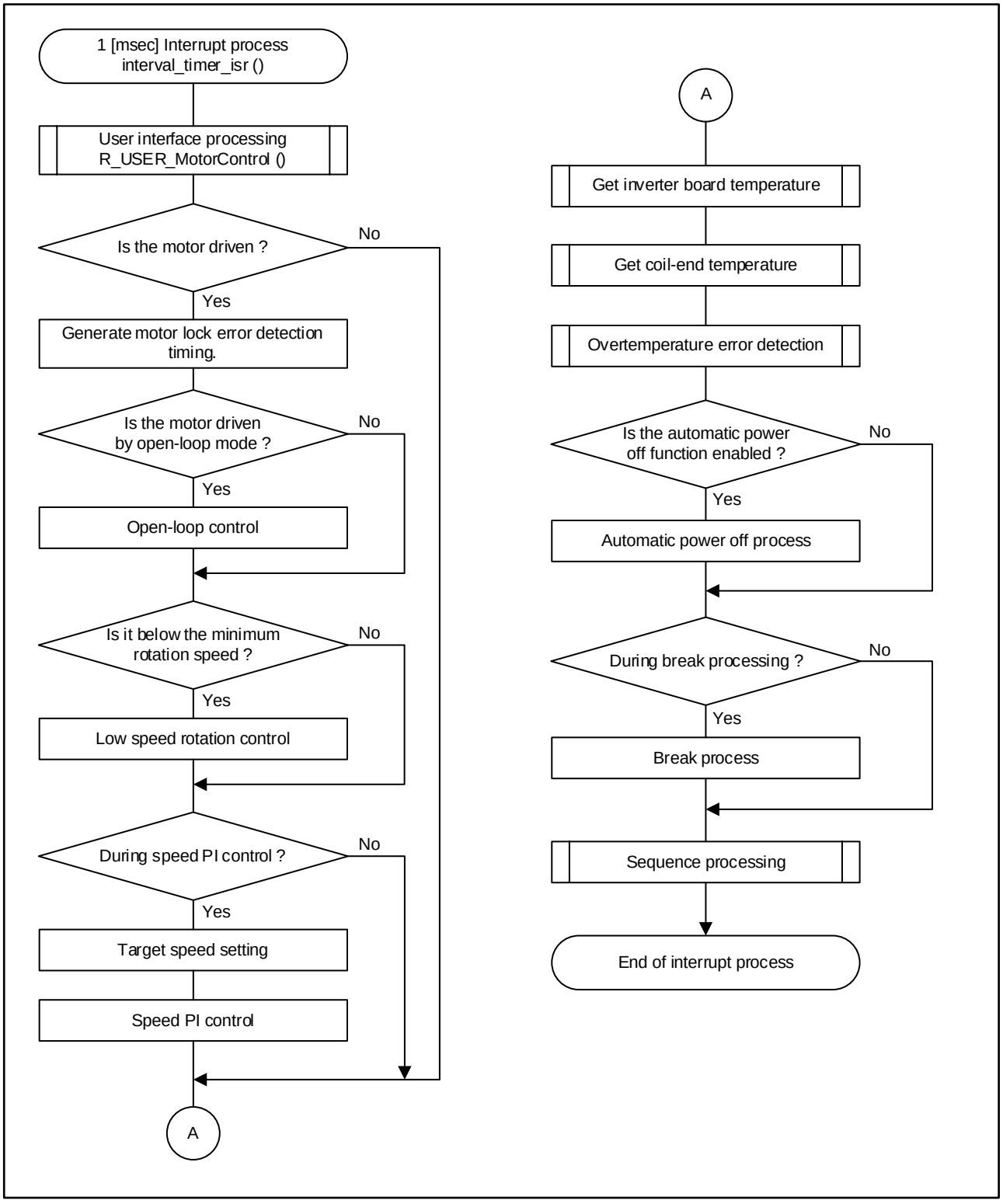

Figure 5-11 Interval Timer Interrupt Handler

5.7.3 User Interface Processing Function

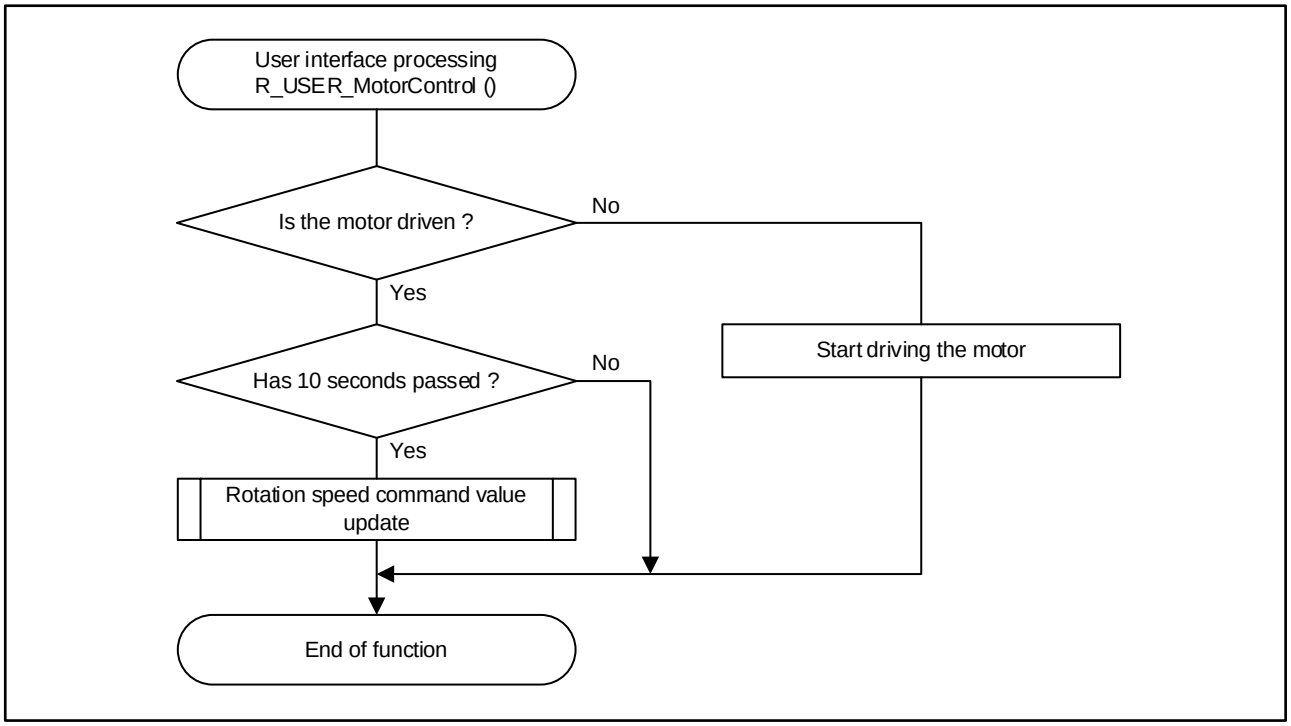

Figure 5-12 User Interface Processing Function

5.7.4 Carrier Frequency Interrupt Handler

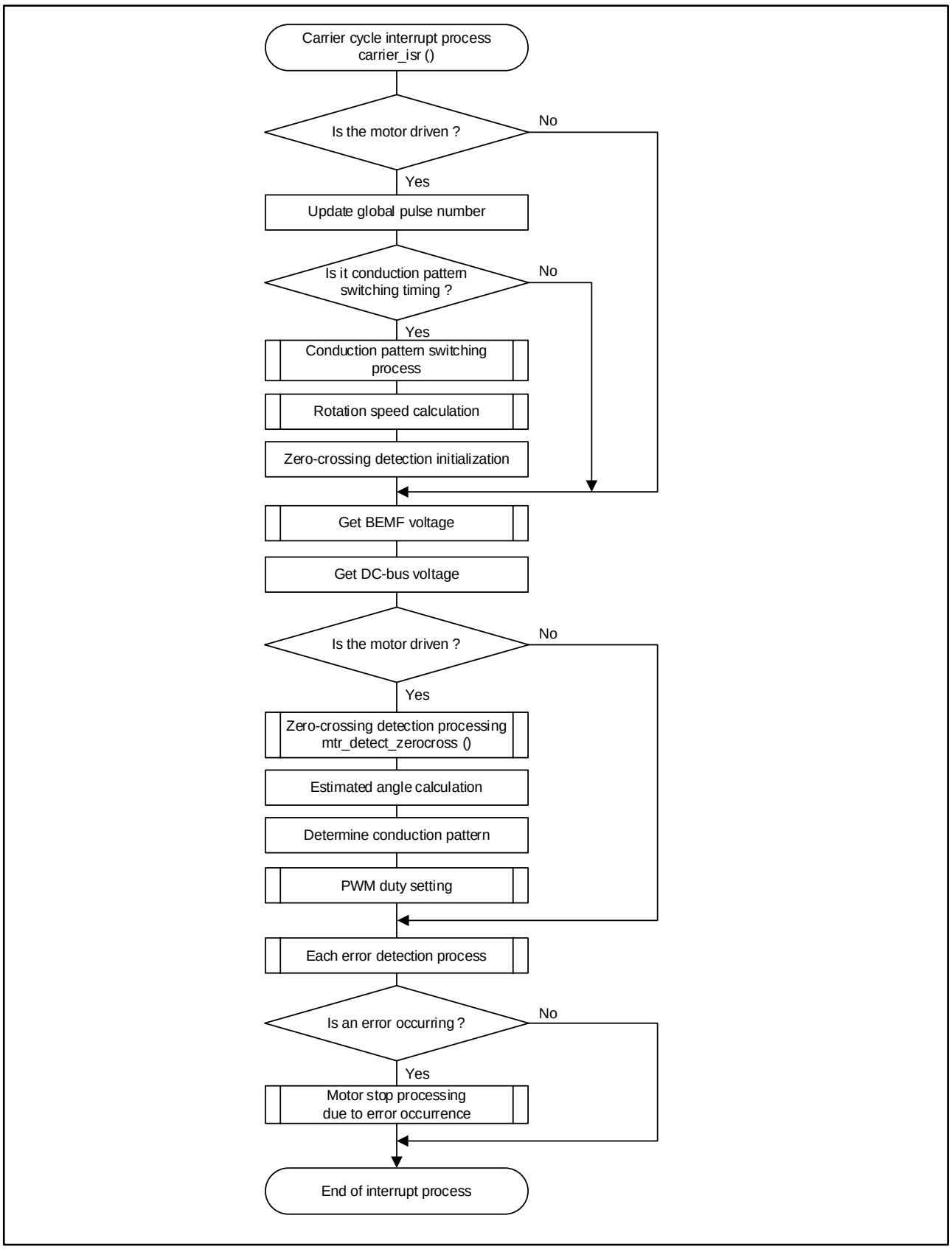

Figure 5-13 Carrier Frequency Interrupt Handler

5.7.5 Zero-crossing Detection Function

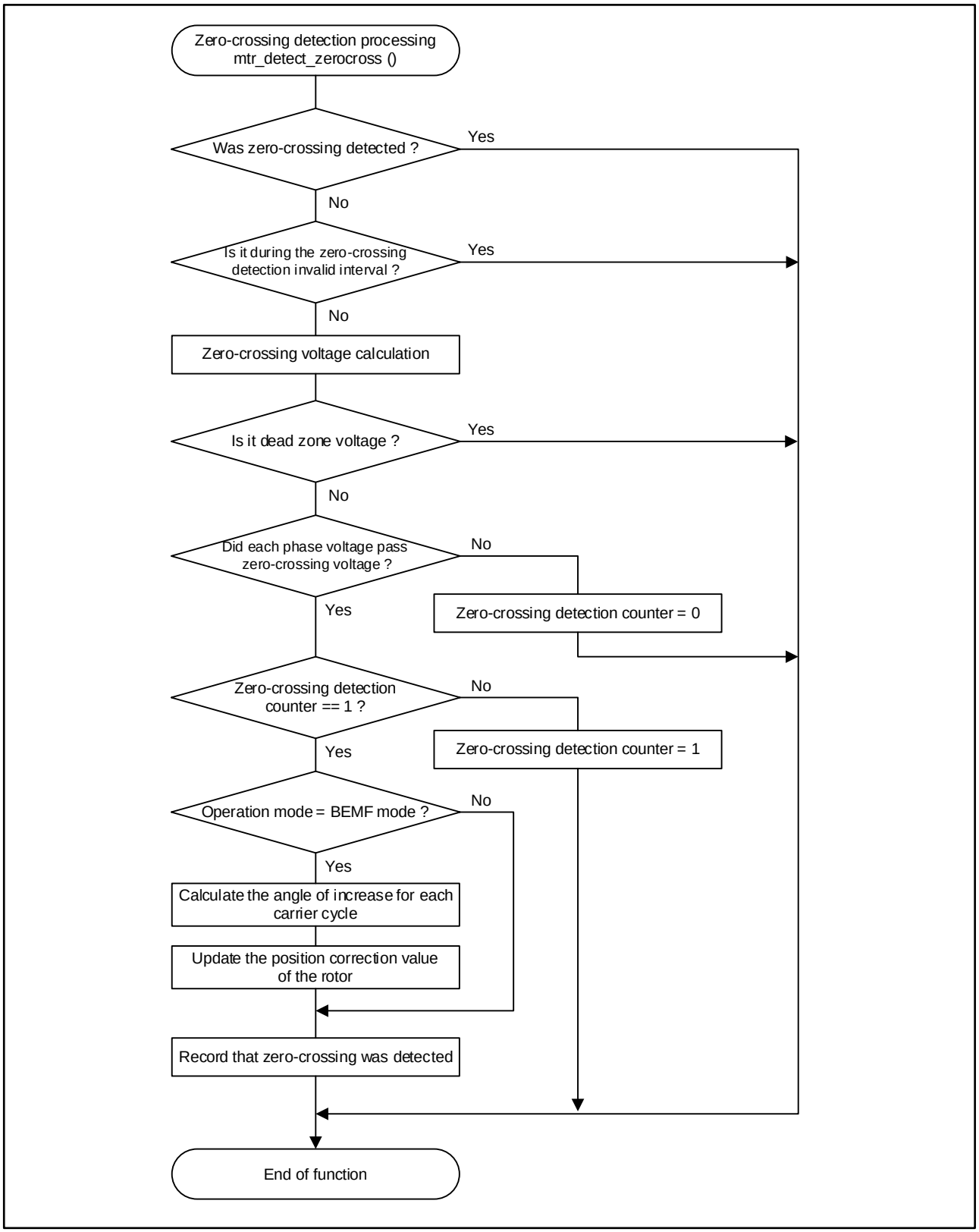

Figure 5-14 Zero-crossing Detection Function

5.7.6 A/D Conversion Completion Interrupt Handler

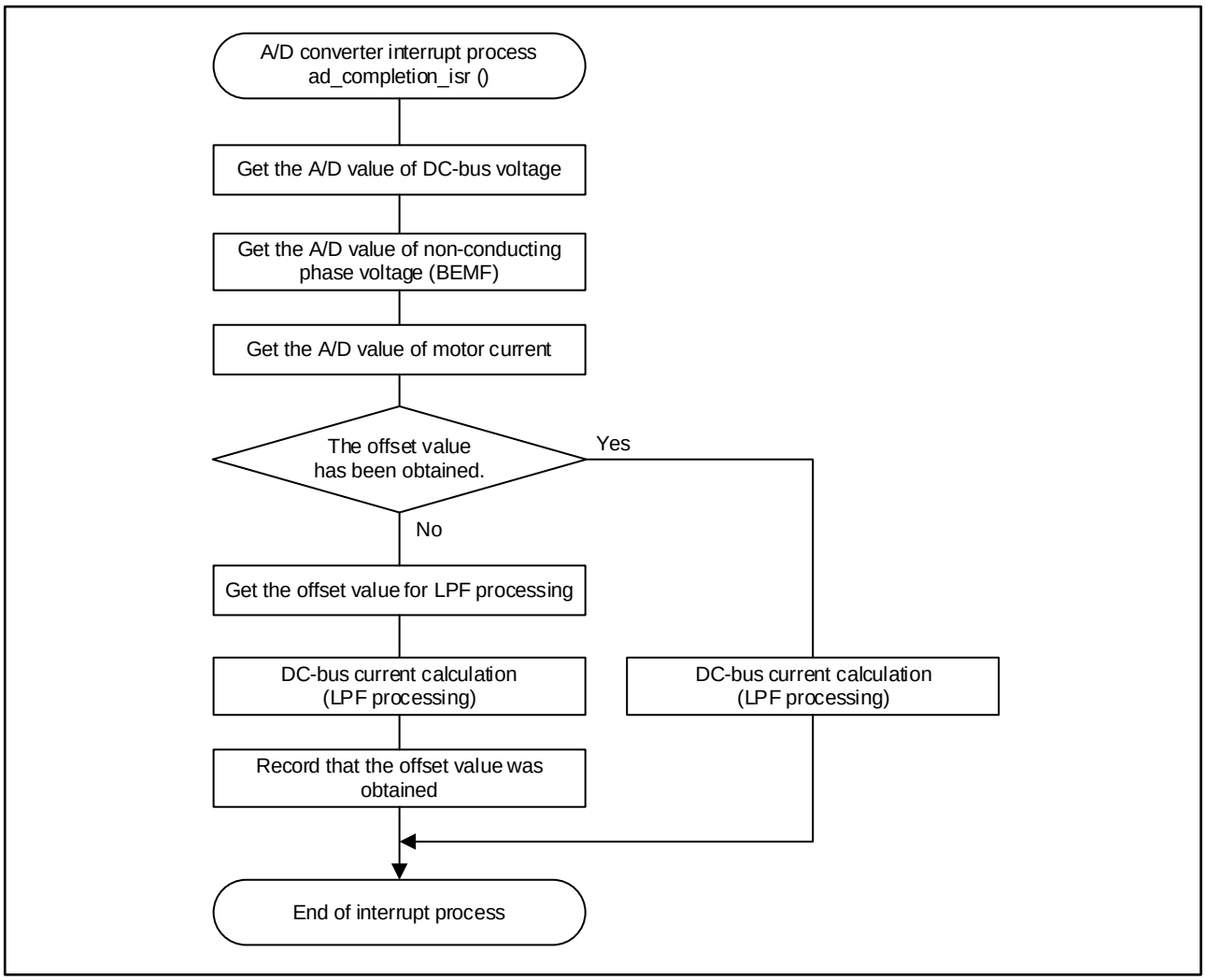

Figure 5-15 A/D Conversion Completion Interrupt Handler

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	2022. 9.30	–	1st edition issued
1.10	2023. 6.30	P.7	Update software version from V.1.04 to V.1.05.
		P.9	Corrected the following typos in item “Processing stop for protection” of Table 2-4. Rotation speed exceeds ~~9000~~ 10000 [rpm]
		P.9	The check timing is described in Table 2-4 “Processing stop for protection” (inverter board and motor coil-end temperature).
		P.34	Added function description of overtemperature monitoring to interval timer interruption (INTTM01).
		P.38	Corrected the variable definition of the following functions from “static variable (s_temp)” to “auto variable (temp)”. • fix8_mul_int16 () • fix10_mul_int16 () • fix12_mul_int16 () • fix14_mul_int16 () • fix16_mul_int16 ()
		P.38	Removed overtemperature detection from function err_check_error_cur_loop ().
		P.38	Added a new function err_check_over_temp ().
		P.39	Added function of overtemperature detection to interval_timer_isr ().
		P.48	Added a new function (Overtemperature error detection) for flowchart of Interval Timer Interrupt Handler.

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

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

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

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

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

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

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

Notice

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

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

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