RX62T

R01AN0901EU0201
Rev. 2.01
Jul. 30, 2014

Single-Shunt Sensorless Vector Control of Permanent Magnet Synchronous Motors (PMSM)

Introduction

This document presents the RX62T single-shunt sensorless vector control solution, which has been implemented on a RX62T evaluation kit with a single-shunt current measurement.

This document describes the evaluation kit hardware platform, single-shunt current measurement, motor phase current reconstruction method, single-shunt sensorless vector control strategy, and software implementation. The sensorless vector control algorithm is based on the method introduced in Renesas Application Note REU05B0103-0100/Rev.1.00.

Usually, three-shunt current sensors are used to measure motor phase currents for current control and flux observer, which results in more hardware cost. In this application note, a complete and unique motor phase current reconstruction method is introduced.

Target Device

RX62T

Contents

1. Overview

Today, cost-effective and high-performance microcontrollers are available, and therefore, many design groups are now interested in implementing sensorless vector control (SVC) of three-phase PMSM. As a result, it has become easy to implement sophisticated, advanced motor control schemes into digitized high-performance motor control systems.

Cost is the main reason why manufacturers are interested in sensorless control methods. Sensorless control methods estimate the speed and angle from the motor currents, thus removing the speed sensor (typically an encoder) and reducing the cost. General formulation of sensorless vector control use three-shunt current sensors adding cost to the overall drive system. When single-shunt current measurement techniques are used to measure the motor phase currents with a special timer unit, the cost of the drive system is further reduced, because one precision shunt resistor is much cheaper than three-shunt current sensors.

This document presents the RX62T single-shunt sensorless vector control solution, which has been implemented on the RX62T evaluation kit with a single-shunt current measurement. It describes the evaluation kit hardware platform, single-shunt current measurement, motor phase current reconstruction method, single-shunt sensorless vector control strategy, and software implementation. The sensorless vector control algorithm is based on the method introduced in Renesas Application Note REU05B0103-0100/Rev.1.00.

The software described in this application note is applicable to the following devices and platforms:

- MCU: RX62T and RX62N
- Motor: Three-Phase Brushless DC (BLDC) and PMSM
- Platform: Renesas Evaluation Kit
- Control Algorithm: Single-Shunt Sensorless Vector Control

2. Specifications and Performance Data

The implementation of the single-shunt sensorless vector control is based on the Renesas evaluation kit and the RX62T MCU. The main specifications are described as following:

- Input Voltage: 24VDC
- Rated Bus Voltage: 24V
- Output Voltage: 24VAC
- Rated Output Power: 120W
- PWM Switch Frequency: 20KHz
- Control Loop Frequency: 10KHz
- Current Measurement: Single Shunt Resistor
- Implementation: FPU
- CPU Bandwidth: 26.5%
- Used Flash Memory: 25.072Kbytes
- Used RAM: 4.397Kbytes
- Used Stack : 336bytes

3. Hardware Platform

The RX62T evaluation board is a single board with an integrated power inverter with the controller. The hardware includes a low-voltage MOSFET power stage, a communication stage, and a RX62T microcontroller based controller as shown in Figure 1.

The board has the following features:

- A complete 3-phase inverter on-board with a low-voltage motor
- 24V external power supply to provide DC bus voltage, 15V and 5V power supply
- Power devices use Renesas low-voltage MOSFETs
- Power rate up to 120 watts
- Supports three-shunt and single-shunt current measurements
- Easily change jumpers from the external amplifiers to the internal PGA
- USB communication with the PC via a H8S2212 MCU
- Graphical User Interface (GUI) used to both modify the motor and control parameters and tune the speed and position control
- Connectors for Hall sensors and encoder connections
- LCD to monitor the operation status
- Supports the standalone mode set by potentiometer and push-buttons
- Supports the second motor drive, signals, and connector for another motor control power stage

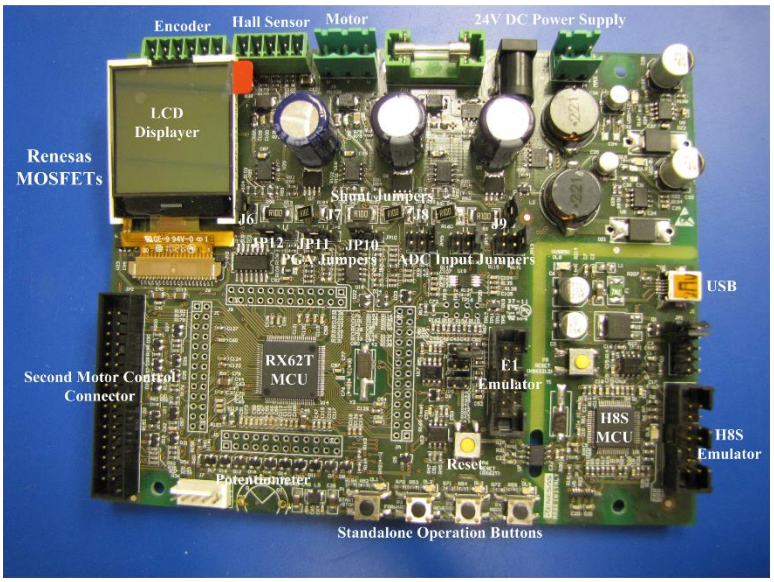


Figure 1 Evaluation Board

4. System Control Block Diagram

The functional block diagram of the single-shunt sensorless vector control is depicted in Figure 2. The system consists of an input AC/DC rectifier, a DC link, and an output DC/AC inverter. The controller to implement the single-shunt sensorless vector control algorithm uses Renesas' RX62T floating-point microcontroller. It can be seen that the three-shunt current measurement is removed. The single-shunt current measurement is adopted instead of the three-shunt current sensing for this solution. Through sampling the DC bus current, the three motor phase currents are reconstructed.

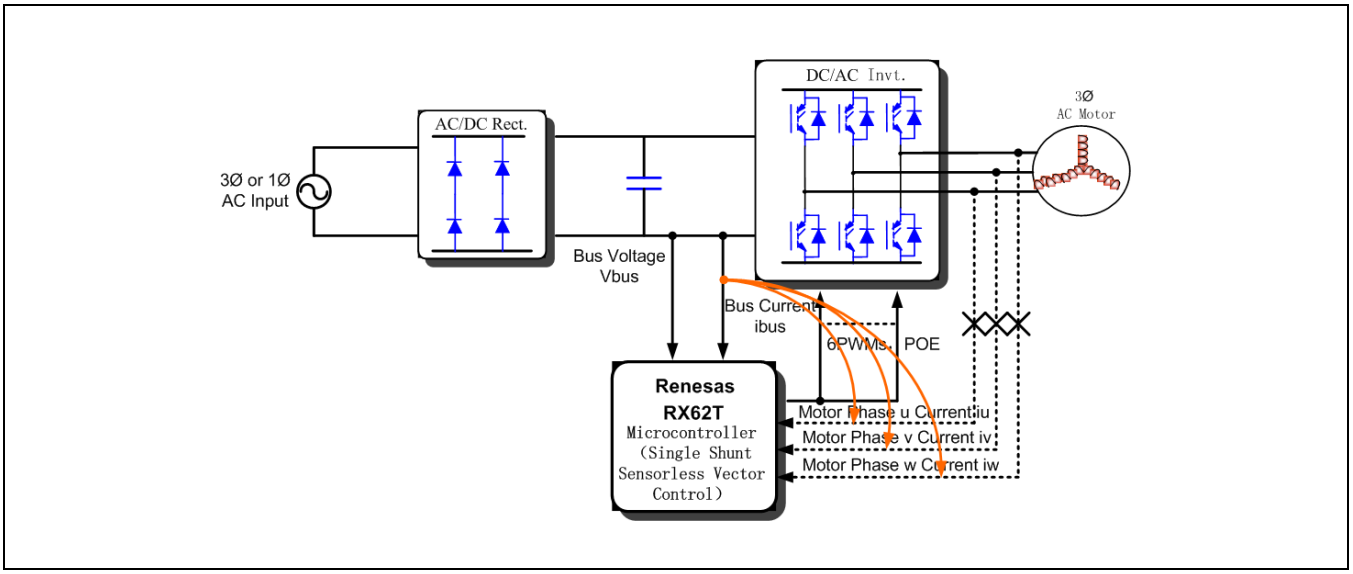


Figure 2 Functional Block Diagram of the Single-Shunt Sensorless Vector Control

The RX62T is a 32-bit high-performance microcontroller with a maximum operating frequency of 100MHz, 165 DMIPS, and a single precision floating-point unit (FPU), which is equipped with multifunction timers (MTU, GPT), high-speed 12-bit ADC, and a 10-bit ADC for facilitating motor control.

Figure 3 shows the block diagram of the RX62T microcontroller for the evaluation kit. The RX62T timer, MTU3 channels 6 and 7 are used to generate 6 PWM signals to drive the on-board motor in the complementary mode. The PWM modulation uses the space vector PWM or the sinusoidal PWM with the third harmonic. The three-phase inverter uses Renesas' low-voltage MOSFETs, which generates three-phase voltages with variable frequency and amplitude to drive the motor to the desired voltage.

The motor currents of i_u, i_v, i_w are measured by three shunt resistors. The currents of i_u, i_v, i_w are measured by the 12-bit ADC unit0 of channel AN002, AN001and AN000, respectively. For the single-shunt current measurement, the current i_w, which represents the DC bus current, is the only input to the ADC channel AN000 by modifying the jumpers on the board. The motor phase currents are then reconstructed by the i_w, and the bus voltage is measured by the 12-bit ADC unit0 with channel AN003.

The speed is given by an external potentiometer, and the speed is input to the 10-bit ADC channel AN1. The MOSFET temperature is measured by the 10-bit ADC channel AN2.

The encoder pulses A and B are input to TCLKA and TCLKB. The Z pulse is input to IRQ0. For the second motor, the encoder pulses A and B are input to TCLKC and TCLKD. The Z pulse is input to IRQ3. MTU3 timer has a phase counting mode to capture two-phase encoder pulse inputs.

When the motor and power board have overcurrent, the current ADC sample circuits generate a port output enable (POE) signal to immediately shut down the PWM outputs from the MCU hardware. In addition, the RX62T control system has various system level safety features to comply with the IEC60730 safety standard such as: low-voltage detection (LVD), independent watchdog timer (IWDT), clock stop detection, ADC self-diagnosis, an output port monitor, etc.

The GUI communicates with the RX62T MCU through USB communication. It can display the motor operation status in real time and tune the motor and control parameters. The board can also run in standalone mode, and the LCD displays the motor status.

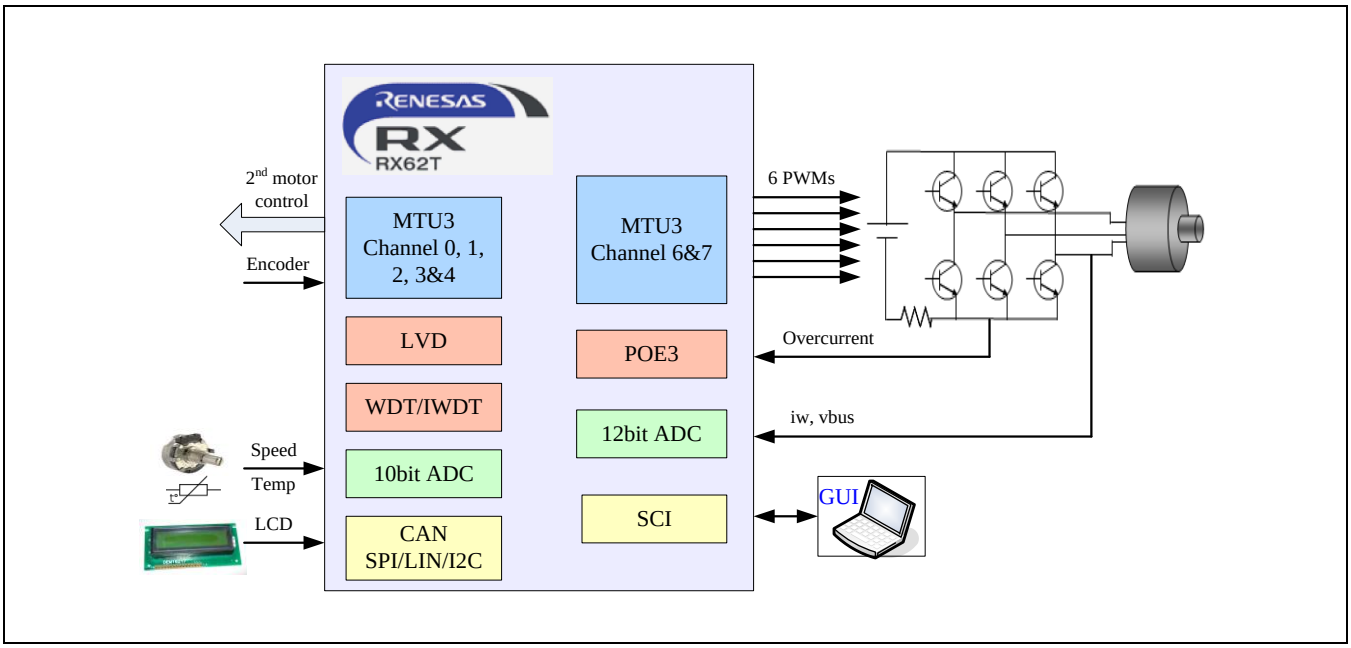


Figure 3 System Control Block Diagram

5. Single-Shunt Motor Phase Current Measurement

5.1 Single-Shunt Current Measurement Circuit

Figure 4 shows the evaluation kit hardware circuit for the single-shunt current measurement. Jumpers J6 and J9 are open, while J7 and J8 are shorted. The composite bus current of the inverter can be measured with a single-shunt resistor of 0.1Ω. Table 1 lists the jumper settings for the single-shunt current measurement.

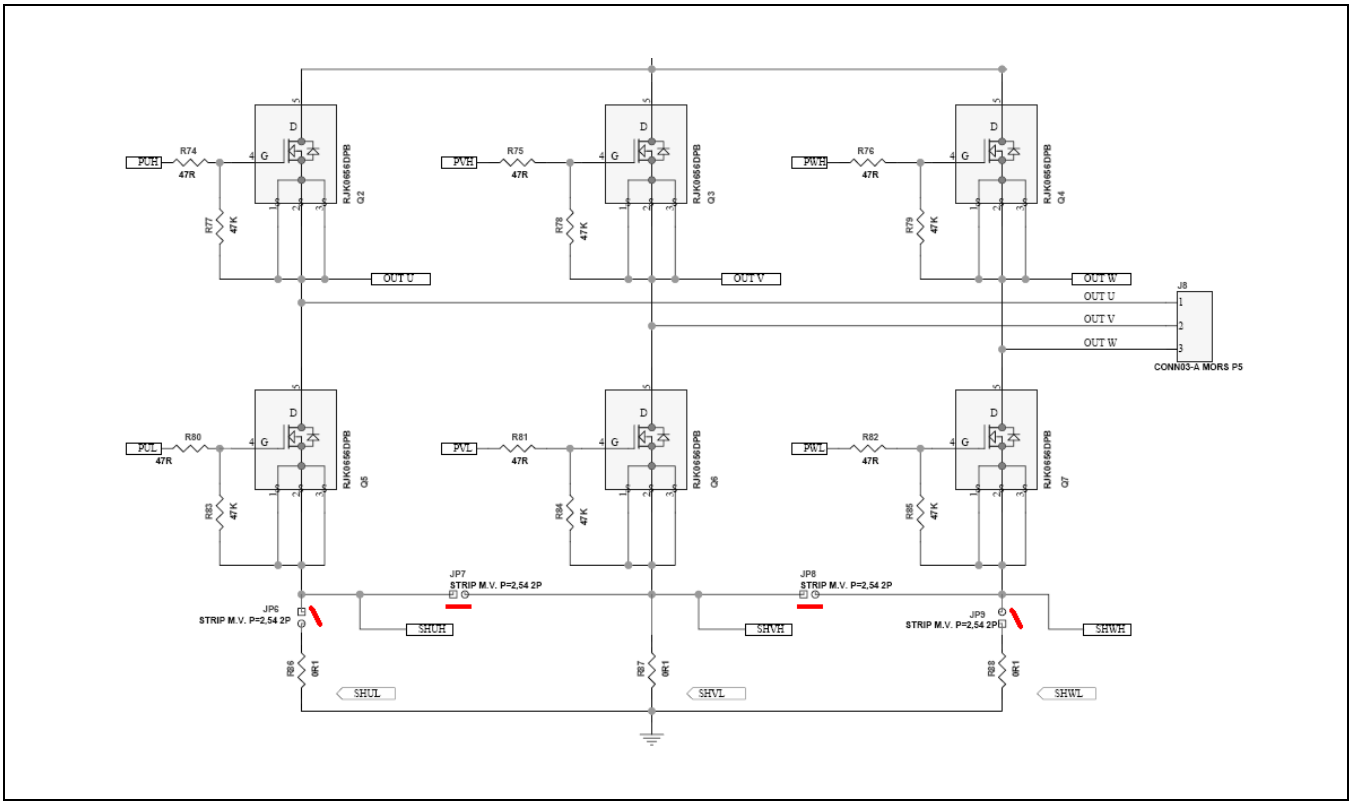


Figure 4 Hardware Circuit of Single-Shunt Current Measurement

Table 1 Jumper Settings for Single-Shunt Current Measurement

Jumper	J6	J7	J8	J9
State	OFF	ON	ON	OFF

5.2 Single-Shunt Current Reading

Figure 5 is the shunt current reading related to the switch pattern.

- When only one or two of the lower switches are ON, the current in the shunt has a relation with the phase currents.
- When only one lower switch is ON, the shunt current is the corresponding phase current.
- When two lower switches are ON, the shunt current is the sum of the corresponding phase currents, so it is equal to minus the third phase current.

It is NOT always possible to read the phase currents in a single-shunt system. A minimum time is requested after an output commutation for reading both the circuit hardware settling time and the sample time of the ADC converter. Therefore, the single-shunt current reading is possible only when there is enough time difference between the different channel commutations in a PWM period, however, the dead-times have a negative influence.

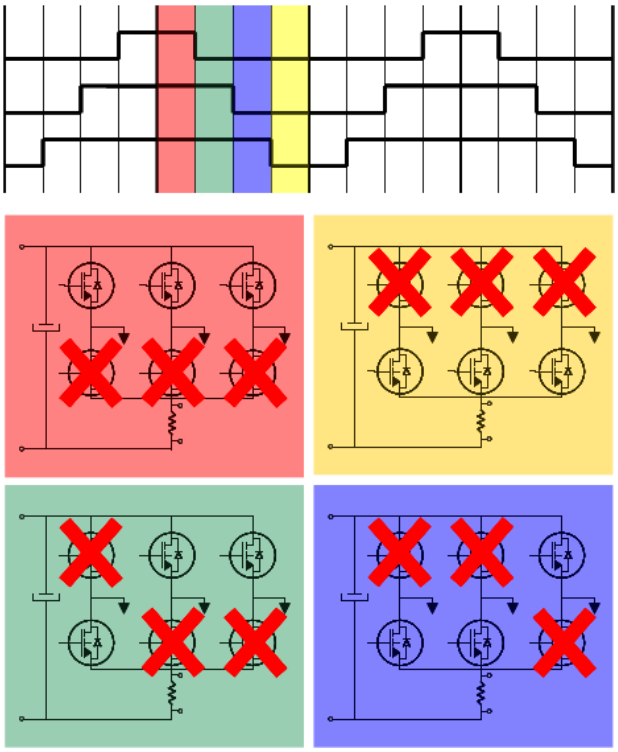


Figure 5 Switch Patterns for the Single-Shunt Current Measurement

5.3 Motor Current Reconstruction from Single-Shunt

The implementation of sensorless vector control with single-shunt current measurement is similar to a two-shunt current measurement, but the motor phase current reconstruction algorithm has to be developed. Figure 6 illustrates how to implement a single-shunt current detection using two MTU3 timer channels of the RX62T microcontrollers MTU7.TADCOBRA and MTU7.TADCOBRB and a hardware triggered ADC channel. Because the RX62T 12 bit ADC channel 0 AN000 has two data registers AN000_IWA and AN000_IWB, one ADC channel is enough for the single-shunt current sampling instead of the traditional method using two ADC channels.

In Figure 6, the W phase has the largest PWM value, V phase has the next smaller value, and U phase has the smallest value. If the shunt current between the rising edges of W phase and V phase is measured, it is concluded that the W phase current is measured. Based on PWM values, only the W phase (that is, only the Wp –upper W phase switch) is on at that time. Therefore, any current measured by the shunt resistor is W phase current only. Next, if the current between the rising edges of V phase and U phase is measured, then, both of the W phase and V phase currents are combined and measured. This also means that the U phase current is measured, because the sum of all three phase currents is zero. Thus, current measurements must be performed at precise times during the PWM interrupt. To trigger the ADC channel and read the ADC data registers of AN000_IWA and AN000_IWB at the precise time, two timer channels MTU7.TADCOBRA and MTU7. TADCOBRB are used. Exact timer counts for these two channels are calculated based on the duty cycle, which is the known output of the control method.

The PWM duty values of all three phases are compared to determine exactly how much time the MTU7.TADCOBRA and MTU7.TADCOBRB timer channels are loaded. It is important to note that the three PWM duty values continue to change constantly, so that the W phase PWM count is not always the largest. Proper flags must be set for the phase every time the largest PWM value is measured for current measurement. All PWM count comparisons, setting of flags for the phase, and corresponding identification for the triggered ADC channels required by this method makes this a complex processing task requiring significantly more code and CPU time. The requirements for CPU bandwidth are generally more than the sensorless vector control with the two shunt current sensor scheme. Fortunately, the ADC channel AN000 is triggered by the hardware, which saves much more ADC conversion time. Furthermore, the RX62T MCU having high computing power makes single-shunt current detection implementation simpler and faster.

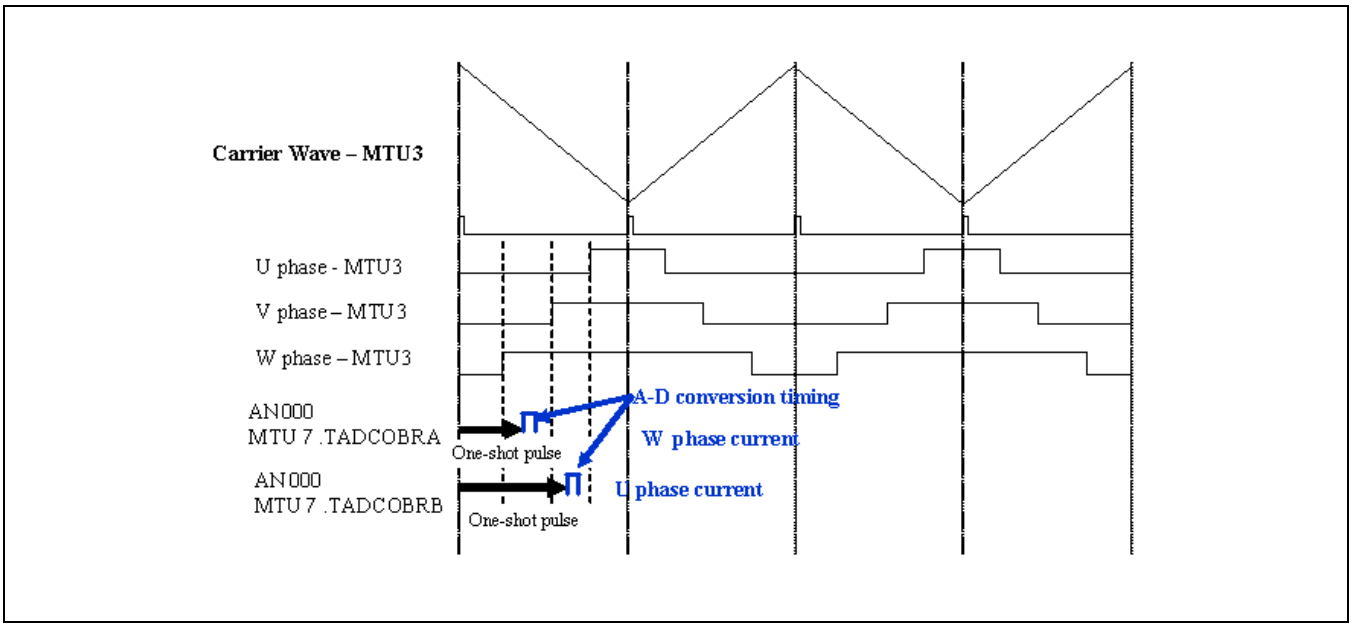


Figure 6 Single-Shunt Current Detection Using Two RX62T MTU3 Timer Channels

The motor phase currents iu, iv and iw are reconstructed from the shunt current. As described above, the shunt current is sampled by AN000_IWA and AN000_IWB, i.e. iaad and ibad, respectively. According to the PWM duty ratios of du, dv, and dw, the AD trigger timers are set, and the motor phase currents are constructed by the following steps:

- If du > dv > dw, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = dv + SS
  MTU7.TADCOBRB = dw + SS
  SS depends on ADC sample time and hardware circuit setting time.

  Convert the single-shunt current into phase u and w currents as:
  ium = iaad - iss_off
  iwm = -ibad + iss_off
  iss_off is the offset of the single-shunt current.

- If du > dw > dv, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = dw + SS
  MTU7.TADCOBRB = dv + SS

  Convert the single-shunt current into phase u and v currents as:
  ium = iaad - iss_off
  ivm = -ibad + iss_off

- If dw > du > dv, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = du + SS
  MTU7.TADCOBRB = dv + SS

  Convert the single-shunt current into phase w and v currents as:
  iwm = iaad - iss_off
  ivm = -ibad + iss_off

- If dv > du > dw, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = du + SS
  MTU7.TADCOBRB = dw + SS

  Convert the single-shunt current into phase u and w currents as:
  ium = iaad - iss_off

  iwm= -ibad + iss_off

- If dv > dw > du, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = dw + SS
  MTU7.TADCOBRB = du + SS

Convert the single-shunt current into phase v and u currents as:
ivm = iaad - iss_off
ium = -ibad + iss_off

- If dw > dv > du, set the trigger time for MTU7.TADCOBRA, MTU7.TADCOBRB as:
  MTU7.TADCOBRA = dv + SS
  MTU7.TADCOBRB = du + SS

  Convert the single-shunt current into phase w and u currents as:
  iwm = iaad - iss_off
  ium = -ibad + iss_off

5.4 Single-Shunt Current ADC Sampling and Scaling

Figure 7 shows the single-shunt current sample and amplifier circuit. Because of the current polarity, the offset should be added to shift the maximum negative current to 0 volts. When the motor operates in the generator mode, the bus current reverses the direction. After the offset and amplifier, the single-shunt current inputs to the RX62T MCU ADC channel AN000 as shown in Table 2.

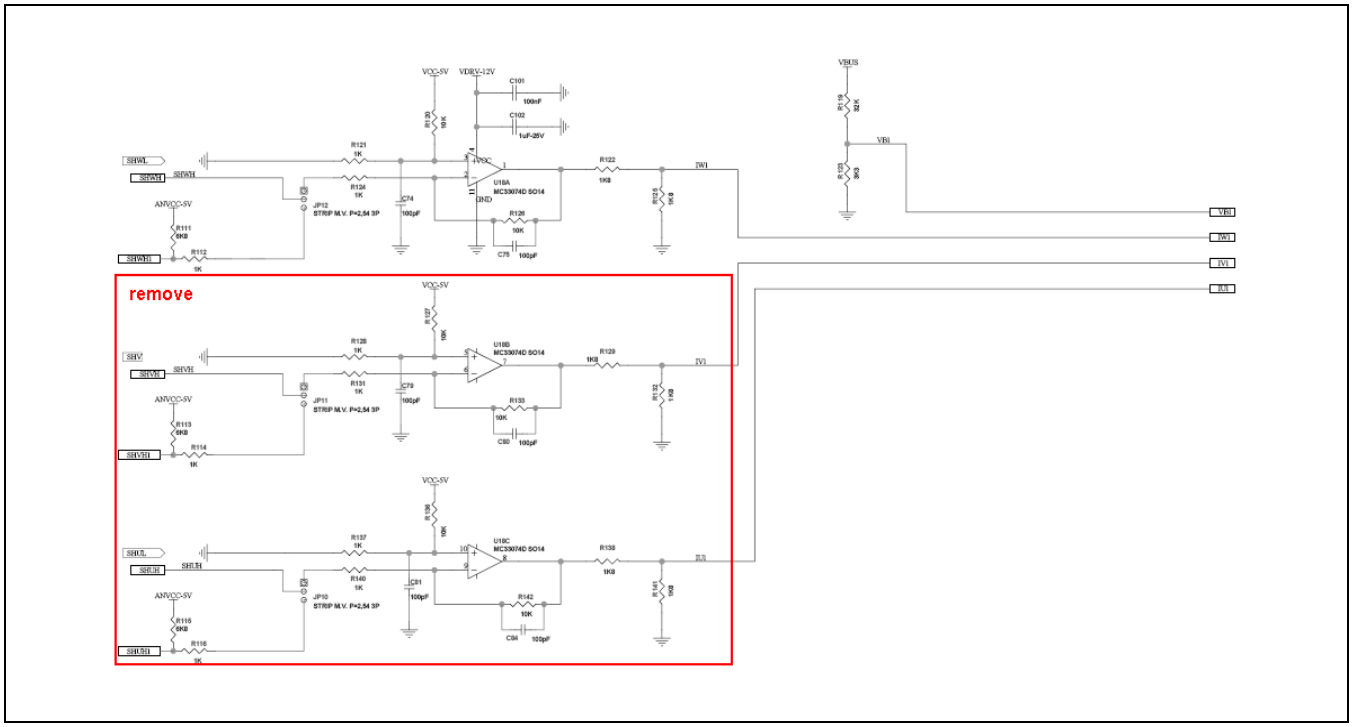


Figure 7 Motor Current Measurement Circuit

Table 2 ADC Conversions

Item	ADC Channel	Conversion ratio (actual value/ADC input value)
DC bus current - iw	AN000	-5A-5A/ 0-5V
DC bus voltage - vbus	AN003	0-50V/0-5V

The shunt resistor used in this application is 0.1Ω. The measured current range is from -5A to 5A, and the gain of the amplifier is set to 5. Figure 8 shows the current sensing scale translation. The shunt resistor 0.1Ω senses the maximum ±5 Amps of current to convert the current to the ±0.5V voltage. The amplifier enlarges this voltage by a factor of 5. The ADC inputs accept the analog input signals in the range of 0-5 volts for the RX62T with the ground referenced to 0 volts. The output voltage of the amplifier is added to the 2.5V offset. The ADC input value of 2.5 V is taken as the current of zero. The ADC input values from 2.5V to 5V correspond to positive current values, and the ADC input values from 0 to 2.5V correspond to negative current values. The 12-bit ADC reads the voltage into the digital value from 0 to 4096.

$$i_{ss} = KADI * (AN000 - i_{ss\_offset})$$

$$KADI = 5/(4096 * Rshunt * Kamp)$$

Where,

i_{ss} is DC bus shunt current;

i_{ss_offset} is DC bus shunt current offset;

$AN000$ is the 12-bit ADC reading value of DC bus shunt current;

$KADI$ is the shunt current scaling;

$Rshunt$ is the value of the shunt resistor;

$Kamp$ is the gain of the current amplifier.

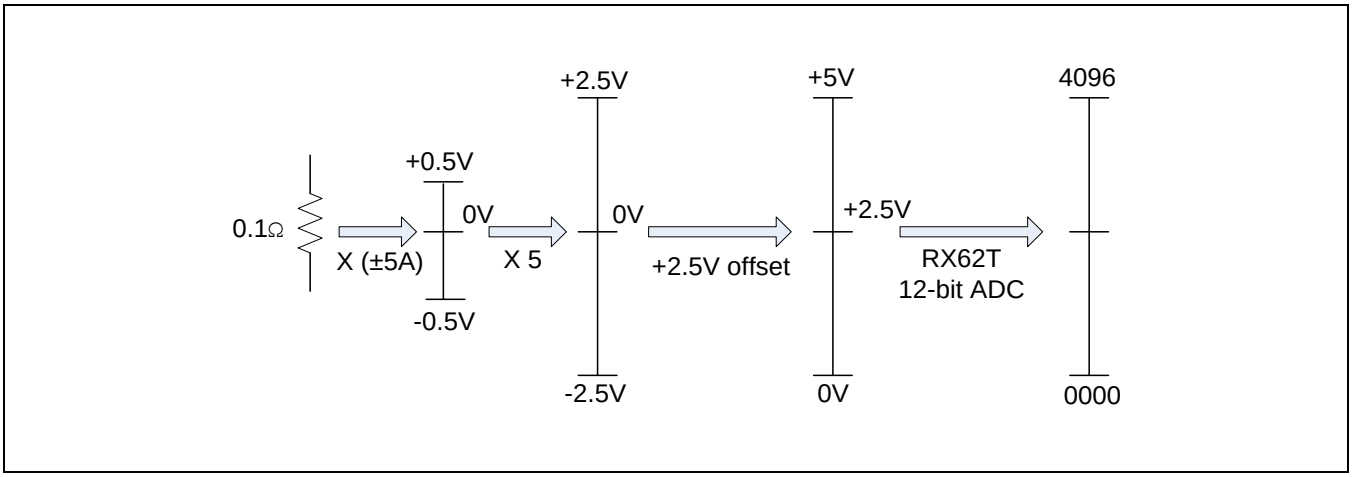


Figure 8 DC Bus Single-Shunt Current Sensing Scale Translation

5.5 Overcurrent Protection

Figure 9 depicts the overcurrent circuit. The switches and gate drivers do not have the hardware overcurrent protection to shut down the PWM signals. The circuit of the overcurrent is added to the evaluation kit. The sampled three-shunt currents or single-shunt current is the input to the circuit as shown in Figure 9. The first part of this circuit is an amplifier, and second part uses the window comparators. When the current is more than -5A or 5A, the comparator generates a low-level signal of POE4. This signal is sent to the POE pin of the RX62T microcontroller.

The RX62T POE can be used to place the complementary PWM output pins for the MTU (MTIOC3B, MTIOC3D, MTIOC4A, MTIOC4B, MTIOC4C, MTIOC4D, MTIOC6B, MTIOC6D, MTIOC7A, MTIOC7B, MTIOC7C, and MTIOC7D) and the large-current output pins for the GPT (GTIOC0A-A, GTIOC0B-A, GTIOC1A-A, GTIOC1B-A, GTIOC2A-A, and GTIOC2B-A) in the high-impedance state in accord with changes in the input signals on the POE0#, POE4#, POE8#, POE10#, and POE11# pins, and to place pin functions multiplexed with complementary PWM output pins for the MTU, pins for MTU3_0 (MTIOC0A, MTIOC0B, MTIOC0C, and MTIOC0D), and pins for the GPT (GTIOC0A, GTIOC0B, GTIOC1A, GTIOC1B, GTIOC2A, GTIOC2B, GTIOC3A, and GTIOC3B) in the high-impedance state in accord with the register settings. It can also simultaneously generate interrupt requests. Furthermore, it is capable of placing pin functions multiplexed with complementary PWM output pins for the MTU, the MTU3_0, and the GPT in the high-impedance state in response to stoppage of the clock signal from the clock oscillator. When the RX62T microcontroller receives the POE signals, it immediately shuts down the PWM outputs without any delay. Therefore, it can protect the power devices and hardware when the overcurrent occurs. Also the POE generates an interrupt to turn off the PWM outputs by software.

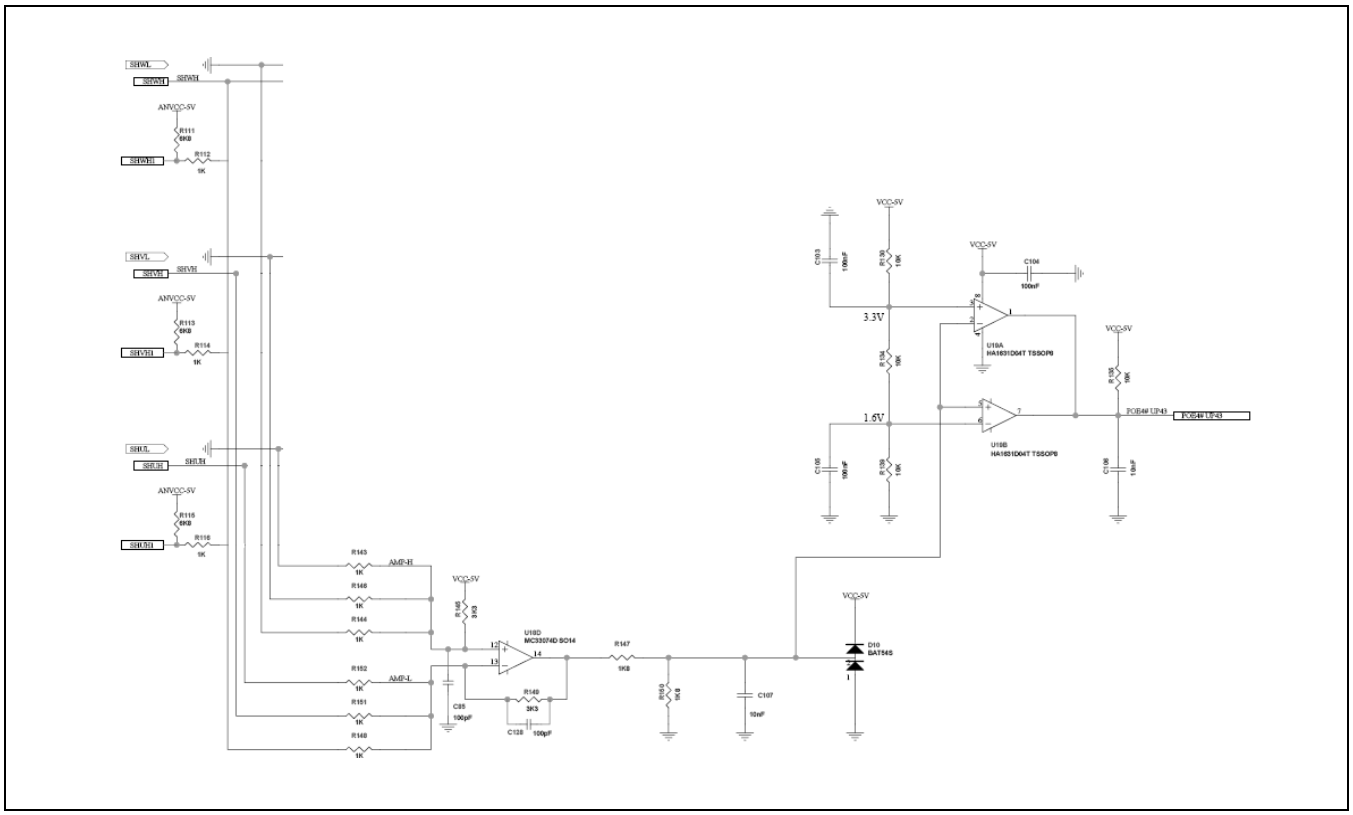


Figure 9 Overcurrent Protection Circuit

5.6 Single-Shunt Current Reading Software Implementation

The software implementation includes the single-shunt current offset calculation MC_SetOff(), single-shunt time setting MC_Set_SS() and the single-shunt current reading MC_Readc_SS().

- MC_SetOff() reads the offset of the ADC channels when the currents are 0, and it is executed once at the beginning.
- MC_Set_SS() routine calculates the exact reading times for the single-shunt ADC conversion according to the commutation timings (duty cycles) du, dv, and dw. Also, it decides if the reading is possible, otherwise a flag is set. The cr_ss variable keeps memory of each particular case.
- MC_Readc_SS() is the routine of the single-shunt conversions reading and management. It is called every interrupt at the trough. It reads the ADC conversion results, and depending on the particular case (indicated by crss), the current readings are converted into internal units and assigned to the correct motor phase currents.

Figure 10 shows the flowchart of the single-shunt current measurement.

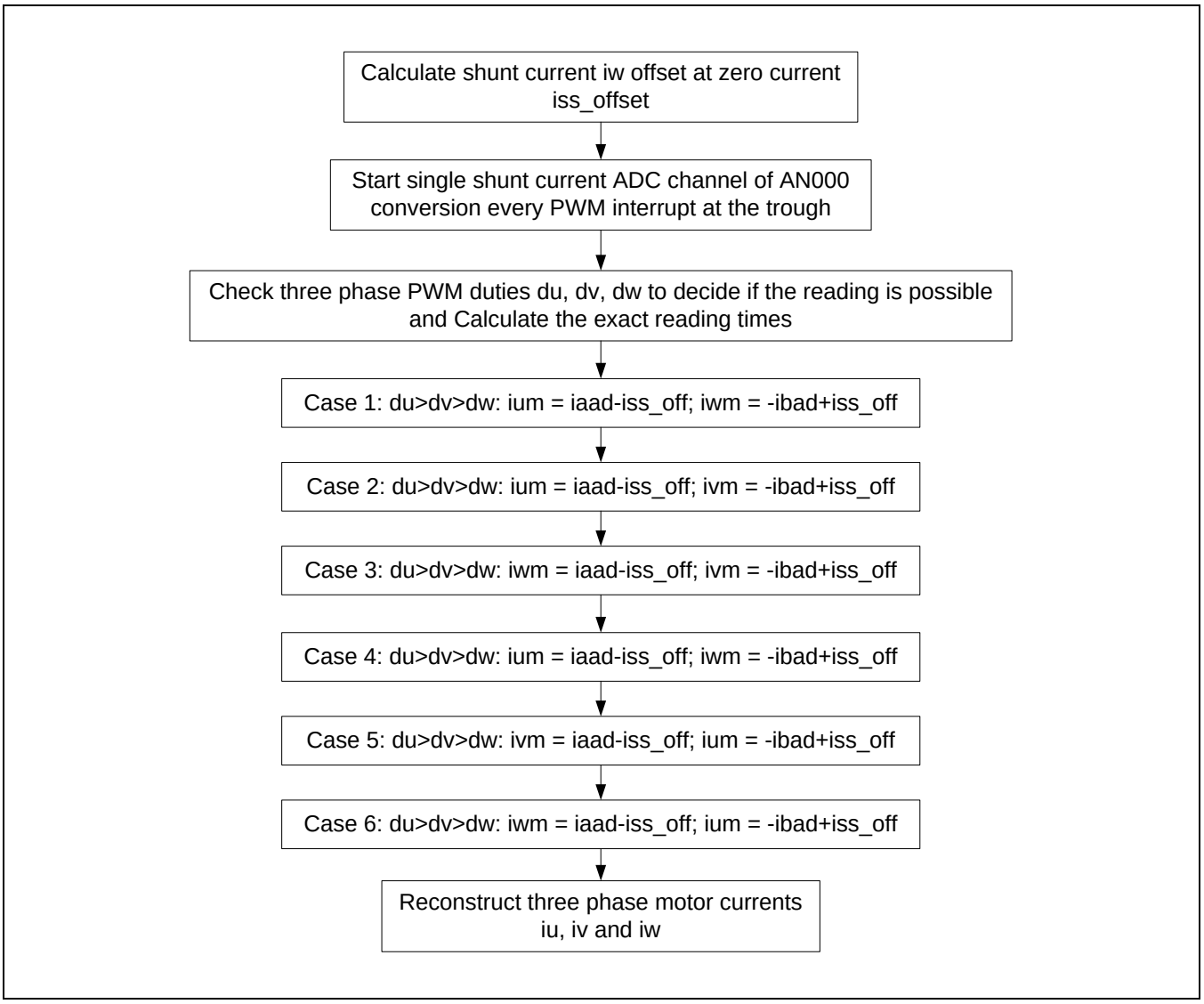


Figure 10 Flowchart of Single-Shunt Current Measurement

6. Single-Shunt Sensorless Vector Control Strategy

The sensorless vector control with single-shunt current measurement method eliminates the speed sensor and three-shunt current sensors. The motor currents are reconstructed using the shunt resistor installed on the low side of the inverter as shown in Figure 4. This shunt resistor is implemented by a precision resistor that is capable of measuring the full current range of the motor. By using single-shunt, three motor currents can be measured. Thus, this method is known as "Single-Shunt Current Sensorless Vector Control".

The implementation is depicted in Figure 11. The overall control system is composed of a flux observer, a speed observer and regulator, a current measurement and reconstruction, two current regulators, VC transformation, and a PWM generator. The speed and current regulators use the traditional PI controller. The overall SVC system includes two control loops, an inner current loop, d-axis current i_d control, q-axis current i_q control, and an outer speed loop. Whenever a reference speed ω^*_r is given, the control system automatically compares it with the motor actual speed ω_r that is directly observed from the flux and currents without the speed sensor. According to the motors mechanical motion equation, the speed error $\Delta\omega_r$ should be the torque profile. Therefore, the output of speed controller could be considered as the torque reference value i^*_{qs}. The rotor flux λ^*_{dr} is supposed to be zero below the rated speed, and it is automatically adjusted above the rated speed when the flux weakening is considered. Once the proper adjustment is done, the motor speed ω_r should follow the given value ω^*_r, and the motor quickly achieves the steady state.

The motor phase currents are converted from the shunt current that is directly measured from the DC bus. The motor current reconstruction block executes the current reconstruction according to the three-phase PWM duty values.

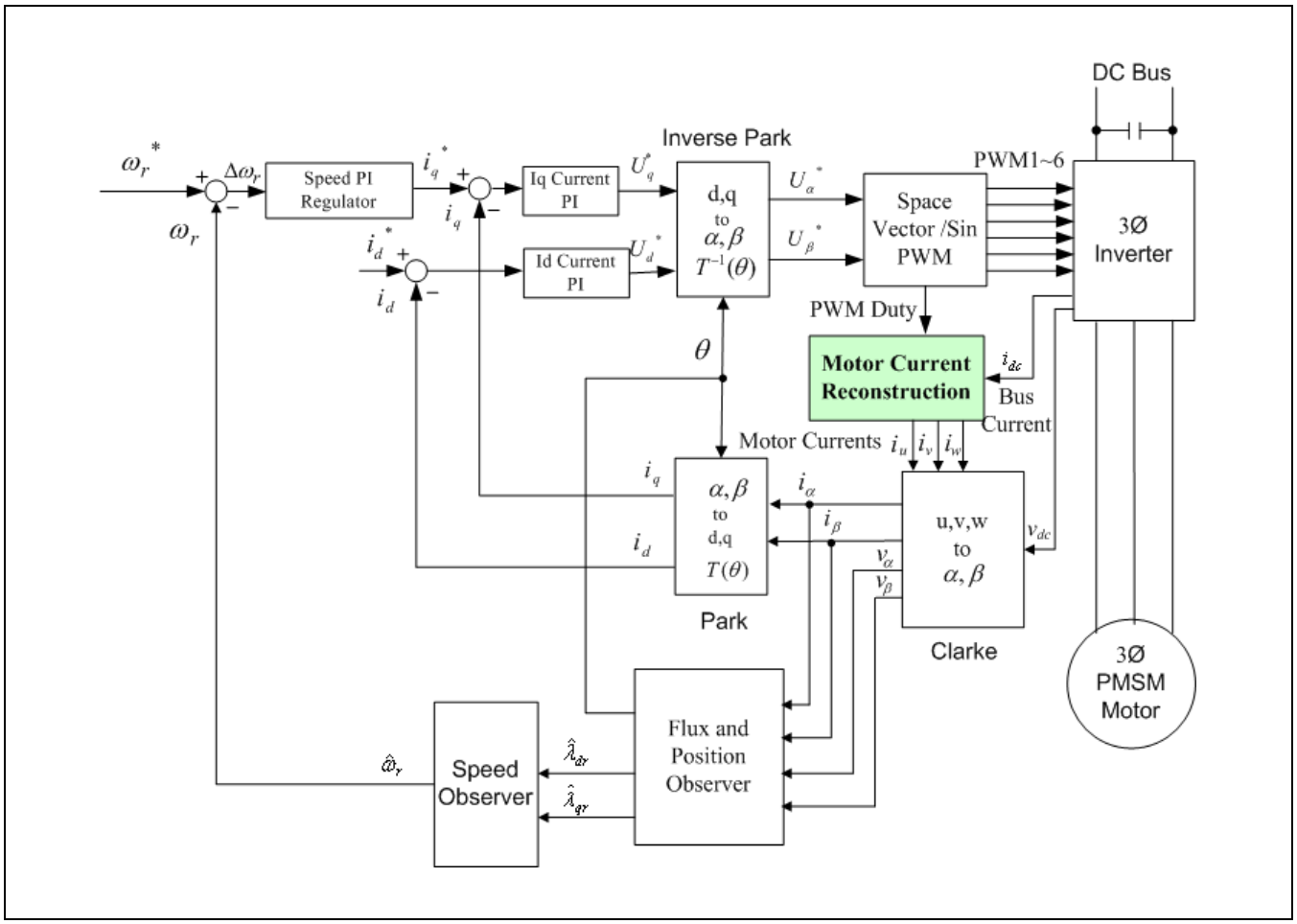


Figure 11 Block Diagram of the Single-Shunt Sensorless Vector Control

7. Software Descriptions

The single-shunt sensorless vector control software has the following features:

- All codes are written in C language.
- The software is modularized according to the SVC block diagram (as shown in Figure 9).
- Core SVC modules can be generally used without any changes.
- I/O definitions and basic MCU drivers are automatically ported by e²Studio.
- Motor and control parameters are easily tuned through a header file of "customize.h" and the GUI.

7.1 Single-Shunt Sensorless Vector Control Software Implementation

The single-shunt sensorless vector control software architecture is similar to the one in Renesas' Application Note REU05B0103-0100/Rev.1.00. Figure 12 shows the workspace of the single-shunt sensorless vector control using Renesas' e²Studio IDE (Integrated Development Environment).

- The code includes: dbsct.c; hwsetup.c, intprg.c; main.c; motorcontrol.c; resetprg.c, userif.c and vectbl.c.
- Single-shunt current reconstruction is put in the "motorcontrol.c".
- Core sensorless vector control modules for vector control transformation of the speed and position observer are put in the library of Single_Shnt_SVC_Lib.lib.

Figure 12 Single-Shunt SVC Software Workspace

7.2 Flowchart of Single-Shunt Sensorless Vector Control

The MTU3 timer interrupt is implemented for single-shunt sensorless vector control. Figure 13 is a flowchart of the single-shunt sensorless vector control. It starts with the open loop, and then switches to the closed speed loop.

The major procedures of single-shunt sensorless vector control include:

- Single-shunt current and DC bus voltage are read and converted.
- According to the PWM duty cycles, the motor phase currents are reconstructed from the single-shunt current.
- When the motor powers on, the startup procedure handles the open loop.
- After the motor powers on, the system switches into the closed speed loop.
- The rotor position and the speed are estimated in sync with the carrier frequency in order to rapidly update the position and the speed.
- The current PI controller outputs of v_d and v_q are transformed back to the three-phase voltages of v_u, v_v, and v_w, which are used to calculate the PWM duty ratios to drive the motor to the desired voltages.

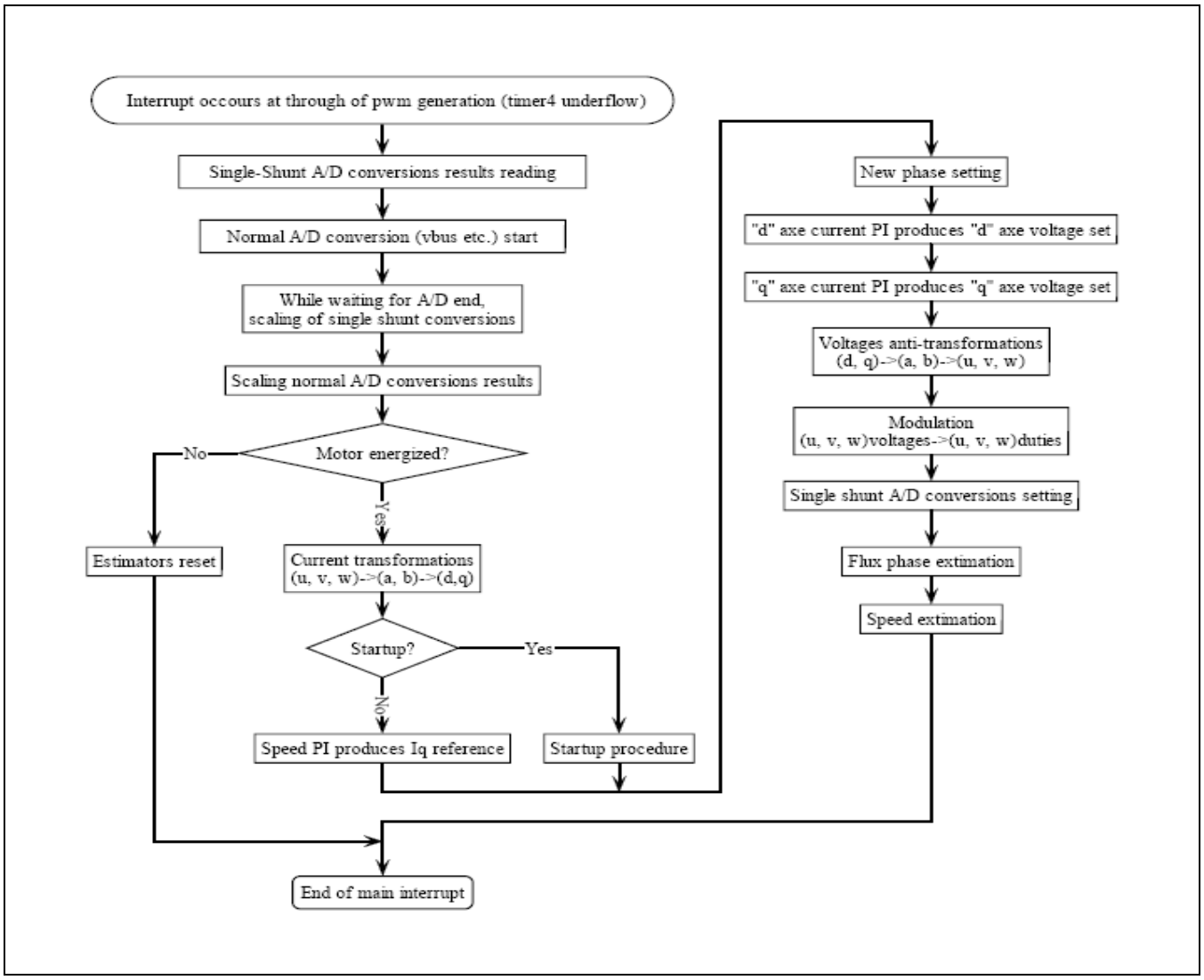


Figure 13 Flowchart of Single-Shunt Sensorless Vector Control

8. Motor and Control Parameter Tuning Example

8.1 Tuning Through the Header File

Shown in Figure 14 is the BLDC motor (BLY171D-24V-4000-1000SI-05), and Table 3 shows the motor's corresponding specifications. The motor is a 24V 4-pole 3-phase BLDC equipped with Hall sensors and a 1000 line per revolution quadratic encoder with index. The rated power is 30 watts, and the rated speed is 4,000 rpm. According to the datasheet, the motor and control parameters have to be properly modified to run the three-shunt sensorless vector control.

Figure 14 The BLDC Motor Included in the Evaluation Kit

Table 3 Motor Specifications

Motor Poles	8
Phase	3
Voltage	24V
Current	1.5A
Power	30 watts
Speed	4000 rpm
Inductance	2.3 mH
Stator Resistor	1.68Ω
Hall Sensors	3
Encoder	1000 pulses/rev

First, define the motor parameters:

- #define R_STA_CUSTOM 8 // stator phase resistance 0.8 Ω *10
- #define L_SYN_CUSTOM 11 // inductance in Henry 2.3mh*10000
- #define POLES_CUSTOM 4 // 4 pairs of poles
- #define I_START_CUSTOM 1.5 // startup current of 1.5A
- #define IQ_MAX_CUSTOM 5.0 // maximum iq current of 5.0A
- #define RPM_MIN_CUSTOM 500 // minimum motor speed of 500 rpm
- #define RPM_MAX_CUSTOM 4000 // maximum motor speed of 4,000 rpm

Second, tune the control parameters:

- #define R_ACC_CUSTOM 1000 // acceleration ramp in 1000 rpm/sec
- #define KP_CUR_CUSTOM 60 // proportional gain of current controller
- #define KI_CUR_CUSTOM 80 // integral gain of current controller
- #define KP_SPD_CUSTOM 40 // proportional gain of speed controller
- #define KI_SPD_CUSTOM 150 // integral gain of speed controller

8.2 Tuning Using the GUI

The motor and control parameters can be tuned through the Renesas GUI as shown in Figure 15. Without modifying the code, the parameters can be set for different motors and applications. There is a parameter window to set up 20 separate parameters. By scrolling up and down through these parameters, the user can decide to make changes to the settings, and "Write" to EEPROM, but this doesn't change the "customize.h" file. The original values will be restored upon clicking

"Reload". From Figure 16, it can be seen that these parameters mirror the #defines in the "customize.h" file, and the motor and control parameters can be easily changed using the GUI.

Figure 15 Evaluation GUI Interface

INDEX	DESCRIPTION	UNIT	MIN	MAX	VALUE	VALID
1	00. Default Parameters Setting	-	0	32767	0	true
2	01. Minimum Speed	rpm	200	5000	500	true
3	02. Maximun Speed	rpm	1000	20000	2500	true
4	03. Acceleration	rpm/s	1	10000	1000	true
5	04. Deceleration	rpm/s	1	10000	1000	true
6	05. Polar couples	-	1	5	5	true
7	06. Startup Current	Apk/10	0	5000	10	true
8	07. Maximum "q" Current	Apk/10	0	5000	20	true
9	08. Stator Resistance	Ohm/10	0	5000	17	true
10	09. Synchronous Inductance	Henry/10000	0	5000	12	true
11	10. Startup Time	ms	300	10000	1000	true
12	11. Current Loop Kp	-	0	2047	60	true
13	12. Current Loop Ki	-	0	1023	80	true
14	13. Speed Loop Kp	-	0	4095	10	true
15	14. Speed Loop Ki	-	0	4095	100	true
16	15. Startup offset V	V/10	0	32767	0	true
17	16. Startup delta V	V/10	0	32767	0	true
18	17. PI Tuning trigger	-	0	32767	0	true
19	18. Free	-	0	32767	0	true
20	19. Free	-	0	32767	0	true

Figure 16 Parameter Window

9. Demonstration Guide

9.1 Introduction to the Demonstration Guide

The purpose of this Demonstration Guide is to help users get up and running quickly with the RX62T motor control kit (YMCRPRX62T). The RX62T microcontroller is pre-programmed to run "Three-Shunt Sensorless Vector Control with External Amplifier". Therefore, the user will need to reprogram the board using the E1 programmer/debugger to demonstrate the Single-Shunt Sensorless Vector Control of PMSM, and later sections will explain how to (1) setup the demo board, (2) build and debug the demo project with e^{2}studio, and (3) run the GUI application. The user needs to connect the motor and the power supply to experience the efficient motor control capabilities of the Renesas RX62T microcontroller.

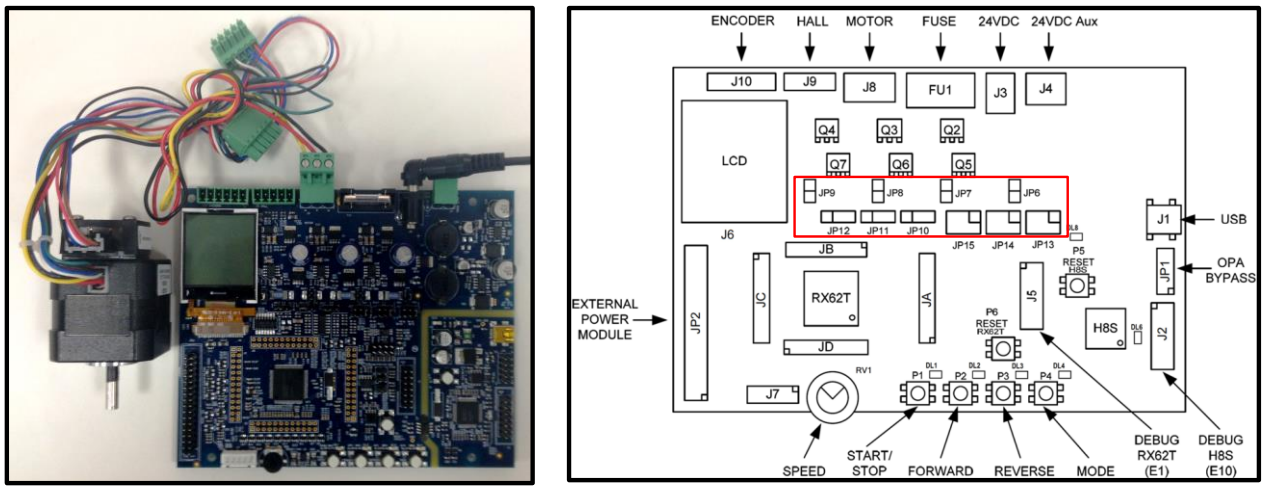
Caution: Do not connect power to the board until all instructions are followed.

The Demo contains the following items:

- RX62T Motor Control Evaluation Board (YMCRPRX62T)
- One BLDC motor with a 3-way Molex connector and encoder cable
- 24V DC power supply
- E1 debugger
- Mini-USB cable
- CD ROM for motor firmware and application GUI

9.2 Demo Board Setup

Figure 17 (a) shows the board with the motor connected to **J8** and the power supply to **J3**. There are four push-buttons, a thumb-wheel potentiometer, a graphic LCD, and a few simple steps to quickly operate the motor out of the box. For debugging or programming, the user needs to connect **J5** with E1. Use the Mini-USB connector, **J1**, in the evaluation board for communication to the GUI.



(a) Board Layout

(b) Functional Layout

Figure 17 Demo Board Layout

Before starting the demo, reconfigure the jumpers (**JP6** – **JP15**) as highlighted with red in Table 4. The jumper's location is shown in Figure 17 (b). The board can be operated in standalone mode or in GUI mode.

Table 4 Configuration Jumper Settings

Operation		JP6	JP7	JP8	JP9	JP10	JP11	JP12	JP13	JP14	JP15
External OPA	3 - Shunt	1 - 2	–	–	1 - 2	**1 - 2**	**1 - 2**	**1 - 2**	**1 - 2**	**1 - 2**	**1 - 2**
	1 - Shunt	**–**	**1 - 2**	**1 - 2**	**–**						
Internal PGA	3 - Shunt	1 - 2	–	–	1 - 2	2 – 3	2 - 3	2 - 3	3 - 4	3 - 4	3 - 4
	1 - Shunt	–	1 - 2	1 - 2	–						

For standalone mode, press and hold **P4** (mode) button during power cycle or **P6** (reset). Then, release the **P4** button. Now the board is in standalone mode. Press **P1** (start/stop) to start or stop motor. Set **RV1** for motor speed and change motor rotation direction by pressing **P2** (forward) or **P3** (reverse) button. **P4** toggles different modes on the LCD.

Note: The user needs to set speed to 2000 rpm or more to run the motor in this demo. GUI mode will be explained in section 9.4.

9.3 Build Project and Debug Operation with e²studio

To generate the firmware program for demonstration, the provided zip file must be imported to the project workspace using e²studio IDE revision 3.0 or higher. The following steps will explain the procedure for importing the project and setting up the debugger in the e²studio IDE.

9.3.1 Build Project Procedure in e²studio

Before importing the project, the user needs to install e²studio version 3.0 or higher and Renesas complier CCRX revision v1.02.01. *Note: This demo will only use Renesas complier CCRX revision v1.02.01.* The user will need to create a new file folder in Windows Explorer. Open the e²studio IDE as shown in Figure 18 and proceed with the following steps:

Step 1. Browse or type the newly created file folder path in the "Workspace Launcher" window and click the <OK> button.

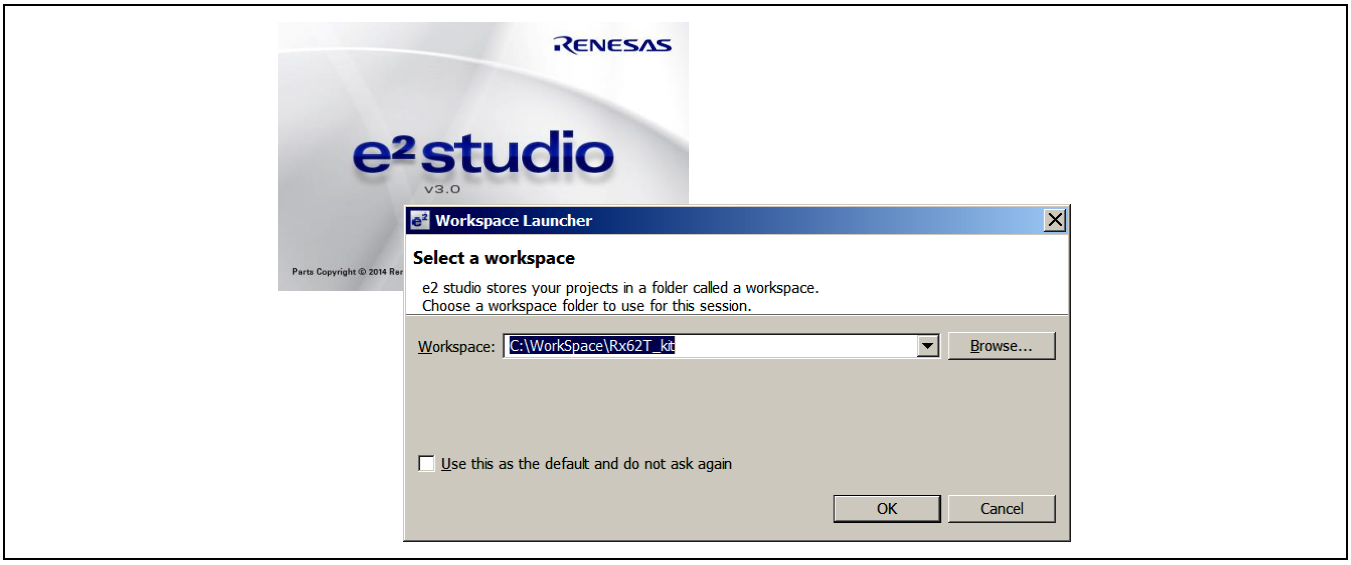


Figure 18 e²studio IDE Start-up Windows and Workspace Launcher

Step 2. Select "Import" from the "File" pull-down menu.

After selecting "Import", Figure 19 shows a *Select* dialog box that prompts the user to "Create new projects from an archive file or directory."

Step 3. Select "Existing Projects into Workspace" from the *Select* pop-up dialog box and click the <Next> button.

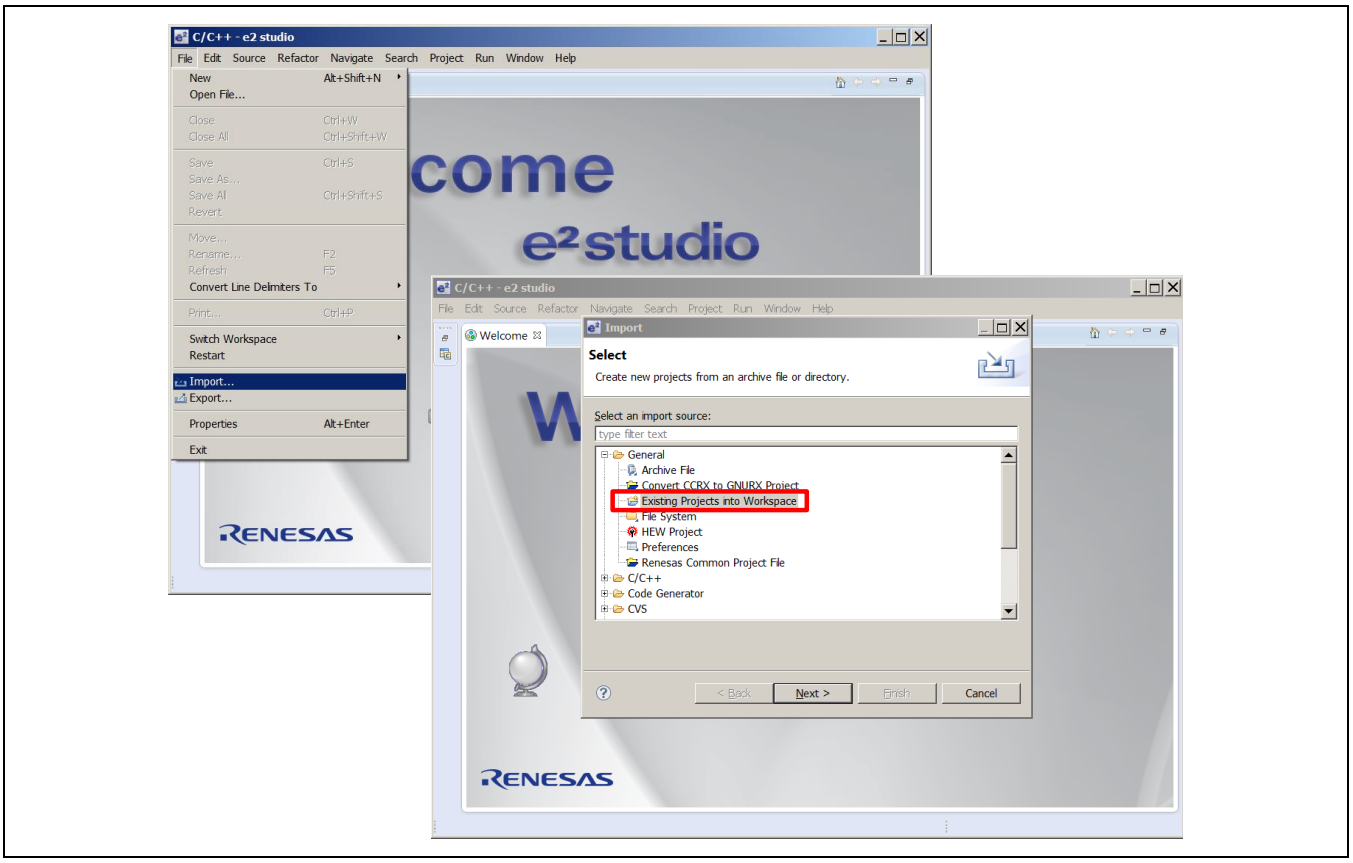


Figure 19 Importing Projects into the Workspace 1 of 2

After clicking the <Next> Button, the *Import Projects* dialog box in Figure 20 prompts the user to "Select a directory to search for existing Eclipse projects."

Step 4. Select the Radio Button "Select archive file" and click <Browse> to locate the Single-Shunt Sensorless Vector Control with Internal PGA zip file to import into the workspace.

The selected project will then appear with a checked box in the *Projects* message box as seen in Figure 20.

Step 5. Check the "Add project to working sets" check box, and then click the <Finish> button to complete the project import.

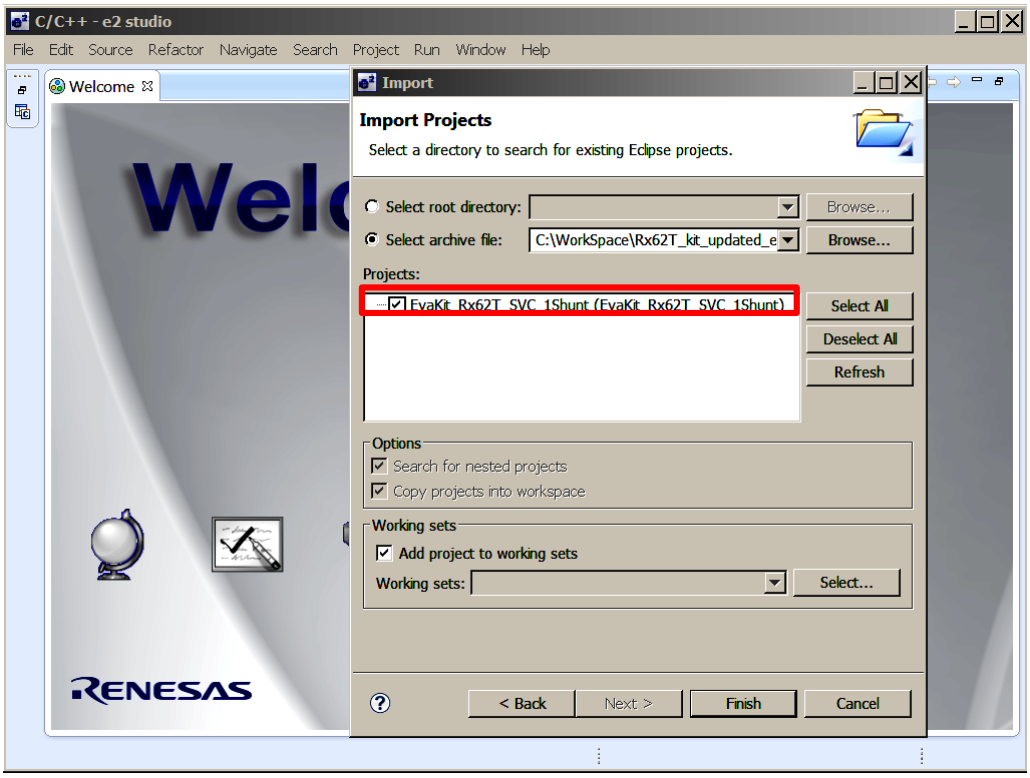


Figure 20 Importing Projects into the Workspace 2 of 2

If the file does not appear with a check box in the *Projects* message box, the selected zip file is the wrong zip file type, or it was not properly exported. If the file already exists in the workspace, then the user will see a message that states, "Some projects cannot be imported because they already exist in the workspace."

After clicking <Finish>, the imported project is now in the e^{2}studio workspace shown in Figure 21, and the project should be in "Debug" mode by default.

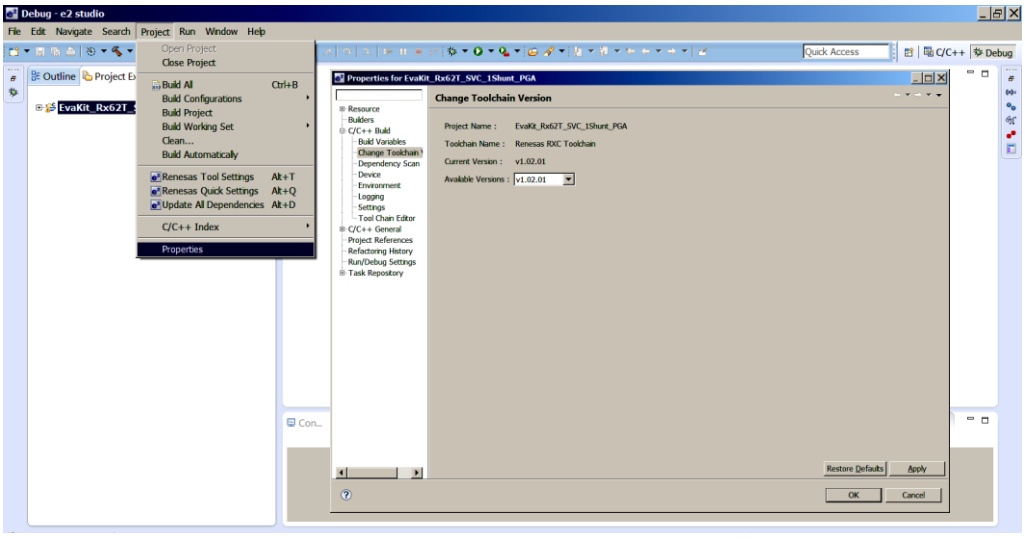


Figure 21 Setting the Toolchain Version in the e^{2}studio Workspace

Step 6. Select "Properties" from the "Project" pull-down menu and expand "C/C++ Build" section. Select the "Change Toolchain Version" option and set the "Available Versions" to v1.02.01.

Step 7. Select the "Clean" command from the "Project" pull-down menu for cleaning and rebuilding the project.

Figure 22 shows the “Clean” Windows dialog box.

Step 8. Check the “Start a build immediately” option and select the Radio buttons “Clean all projects” and “Build the entire workspace.” Then click the <OK> button to clean and rebuild all projects in the workspace.

For debugging, the target firmware (.x file) is generated in the “Binaries” workspace folder shown in Figure 23. For release, set the active project to release mode for building projects. The target firmware (.mot file) is generated in the workspace “Release” folder.

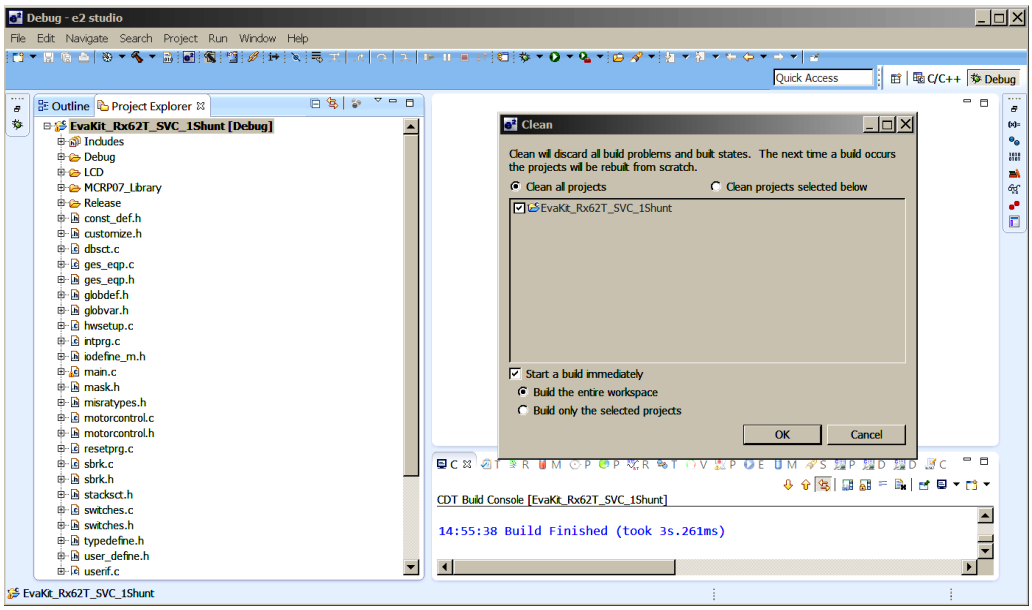


Figure 22 Clean Message Box

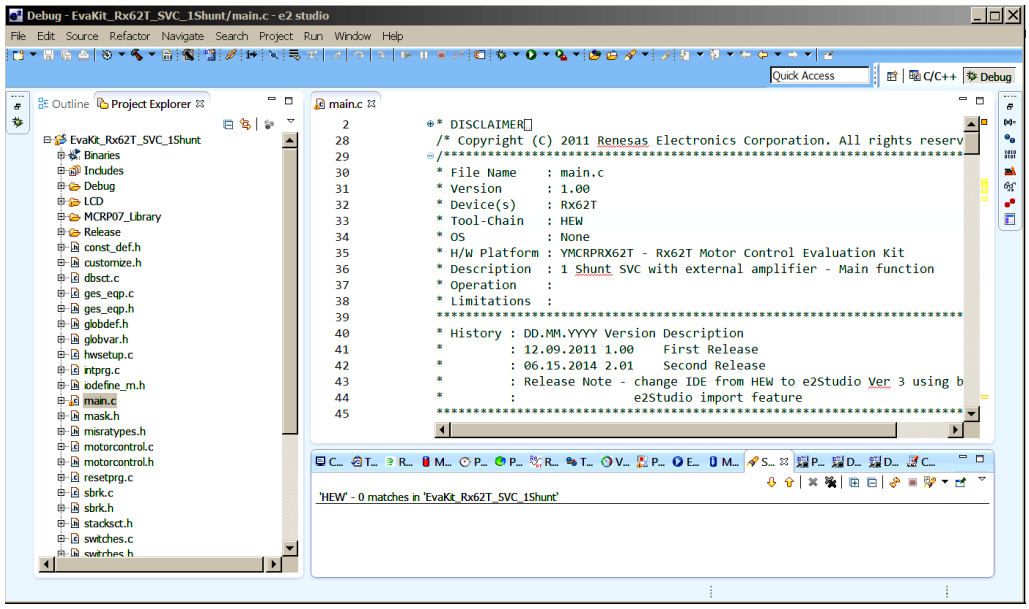


Figure 23 Target Firmware in Workspace

9.3.2 Debug Procedure in e²studio IDE

After generating the target firmware, the user is now ready to setup the debug interface through the E1 debugger. The E1 debugger is necessary as an interface from the software to the hardware. Even if there is no need for any "debugging," this procedure is still necessary to reprogram the board using the provided algorithms. Connect the 24V DC power to **J3**, the E1 Debugger to **J5**, and the motor to the **J8** connector. The connections are shown in Figure 24. Check the recommended jumper settings for this demo (refer to section 9.2).

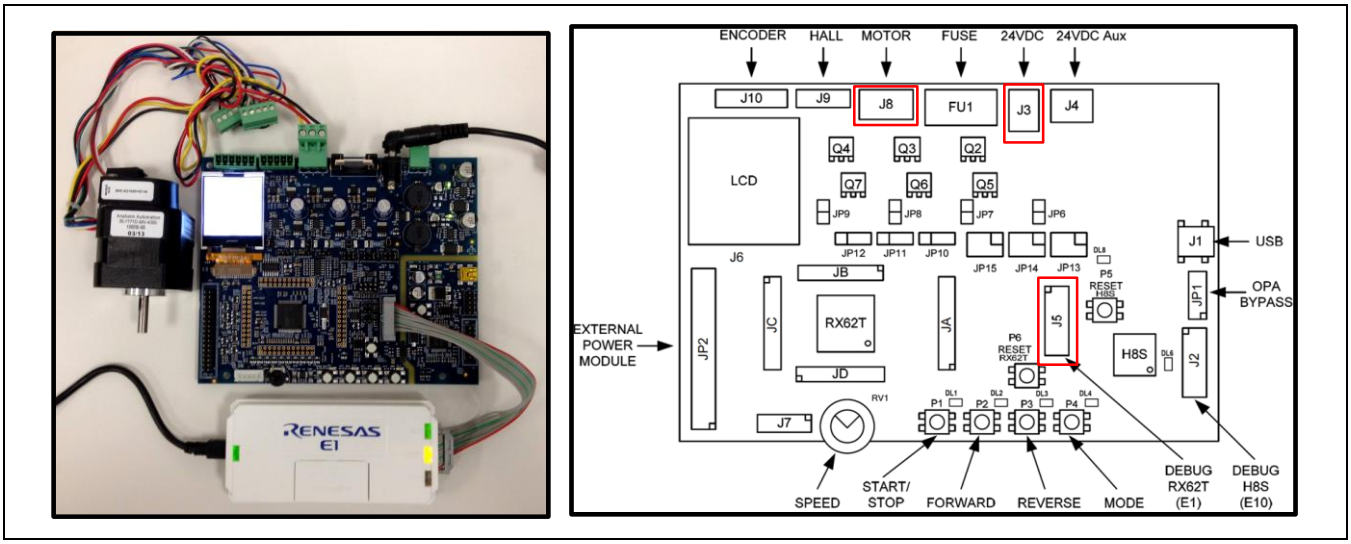


Figure 24 Debug Setup for Demo Board

Step 1. Select "Debug Configurations" from the "Run" pull-down menu or click the debug icon [ ] and select "Debug Configurations"

Now, the user will view the "Debug Configurations" Windows dialog box, as shown in Figure 25.

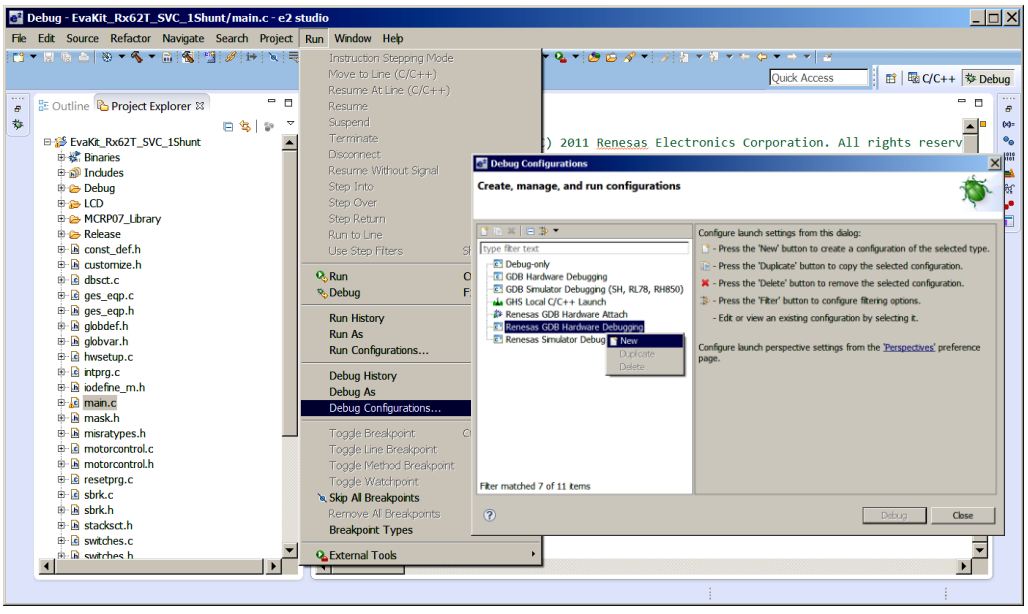


Figure 25 Setup Debug Configuration in Workspace

Step 2. Select "Renesas GDB Hardware Debugging". Using the mouse, right click on "Renesas GDB Hardware Debugging" and select "New" as shown in Figure 25.

Step 3. In Figure 26 under the "Main" tab in Debug Configurations, Select the Single-Shunt Sensorless Vector Control (EvaKit_Rx62T_SVC_1Shunt) as the "Project" and verify the "Build Configuration" tab is selected as "Debug" and the "Use workspace settings" is selected.

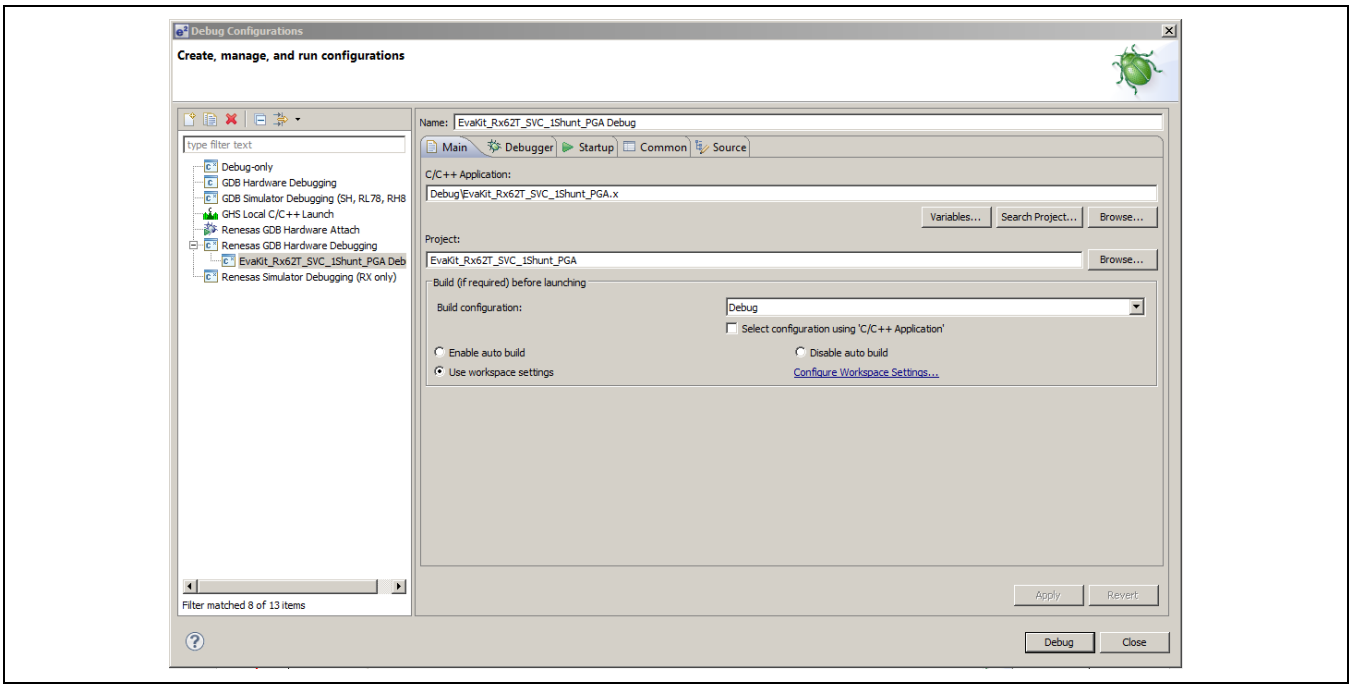


Figure 26 Debug Configuration Main Dialog Box

Step 4. Select the "Debugger" tab as shown in Figure 27.

Step 5. Select the "GDB Settings" sub-tab under the "Debugger" tab and set the "Debugger hardware" to "E1" and "Target Device" to "R5F562TA".

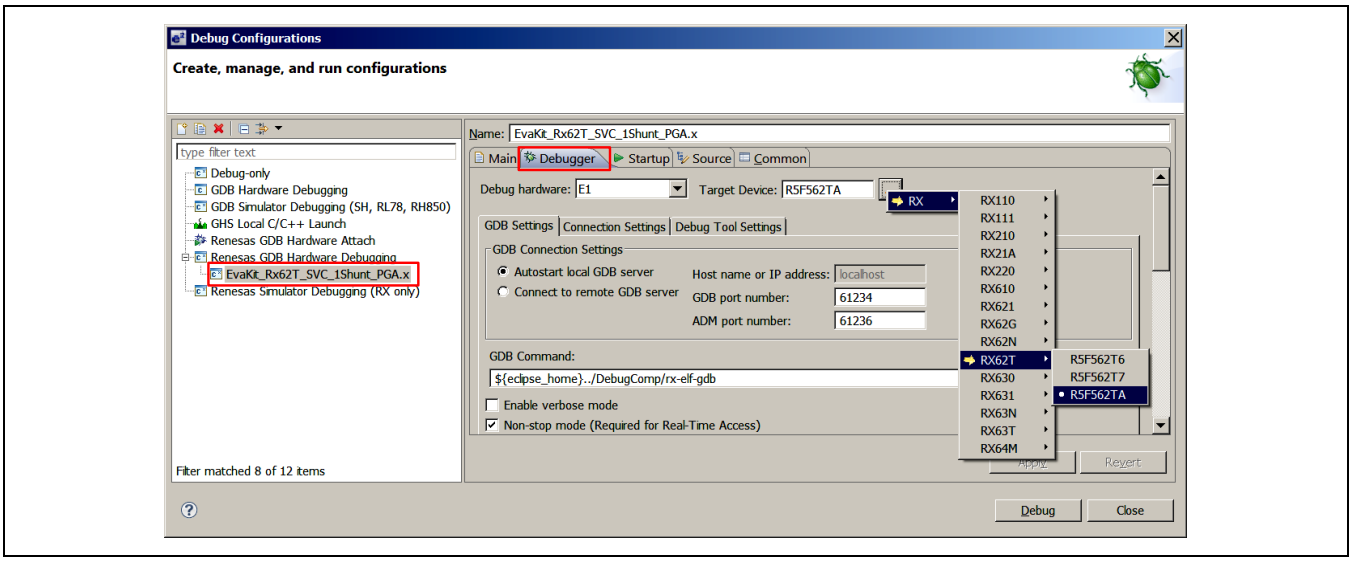


Figure 27 Debug Configuration Debugger Dialog Box

Step 6. Select the "Connection Settings" sub-tab and change the "External Frequency" value to 12.00 MHz and "JTag Clock Frequency" to 12.38 MHz. Set "Power" to "No."

Step 7. Select the "Debug Tool Settings" sub-tab under the "Debugger" tab and select "Big Endian" in the "Endian" setting under "Memory." Then click the <Apply> button.

The typical debug settings for this demo are shown with RED boxes in Figure 28 and in Figure 29.

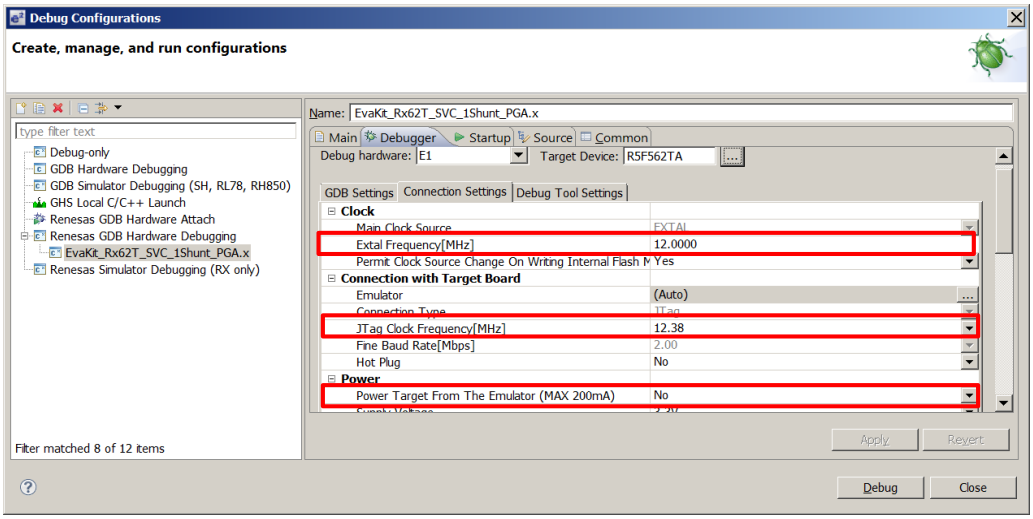


Figure 28 Debug Configuration Dialog Box 1 of 2

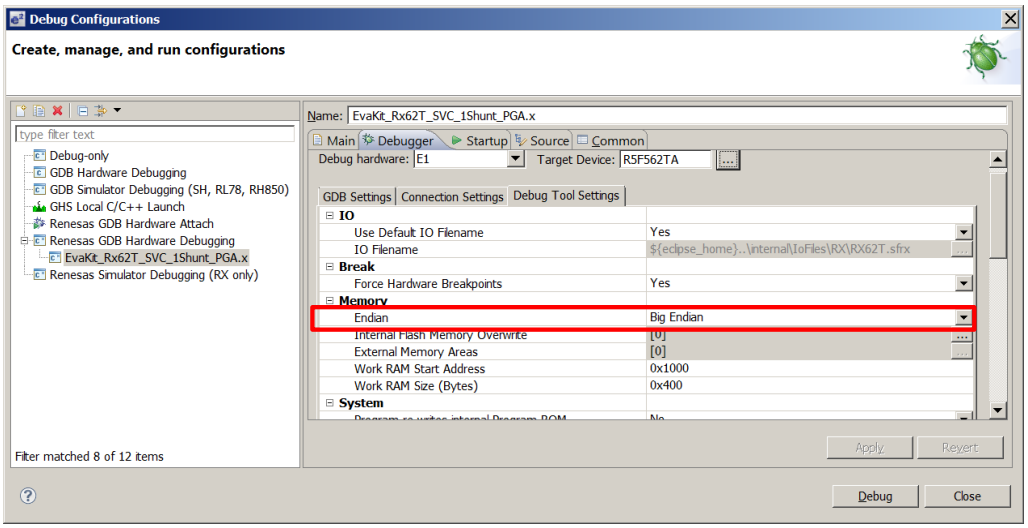


Figure 29 Debug Configuration Dialog Box 2 of 2

Step 8. Check the target board power is ON and verify the connections through the PC, E1 debugger, and the target board.

Clicking the <Debug> button in the "Debug Configurations" dialog box will download the firmware to the target board.

Step 9. Click the "Resume" icon [ ] or press the F8 Key to run the program. This may require the user to press the "Resume" icon or F8 multiple times depending on the delay in the code. The icon should turn gray when the program is running.

The LED **DL1** will blink at about a one second rate continuously while running the target board.

9.4 GUI Operation

This operation requires the demo board to be connected to the PC using the supplied Mini-USB cable.

Step 1. Connect the Mini-USB cable to **J1** from the PC.

LED **DL8** is on when the USB bus power is applied to evaluation board. The PC will recognize the new hardware and will launch the driver installation screen. Follow the instruction from the "Message Box" to install respective USB device driver.

Note: Separate instructions for the USB device driver installation are provided in the Quick Start Guide or the driver will install automatically depending on the CD ROM installer. Figure 30 shows the necessary connections and LED designations.

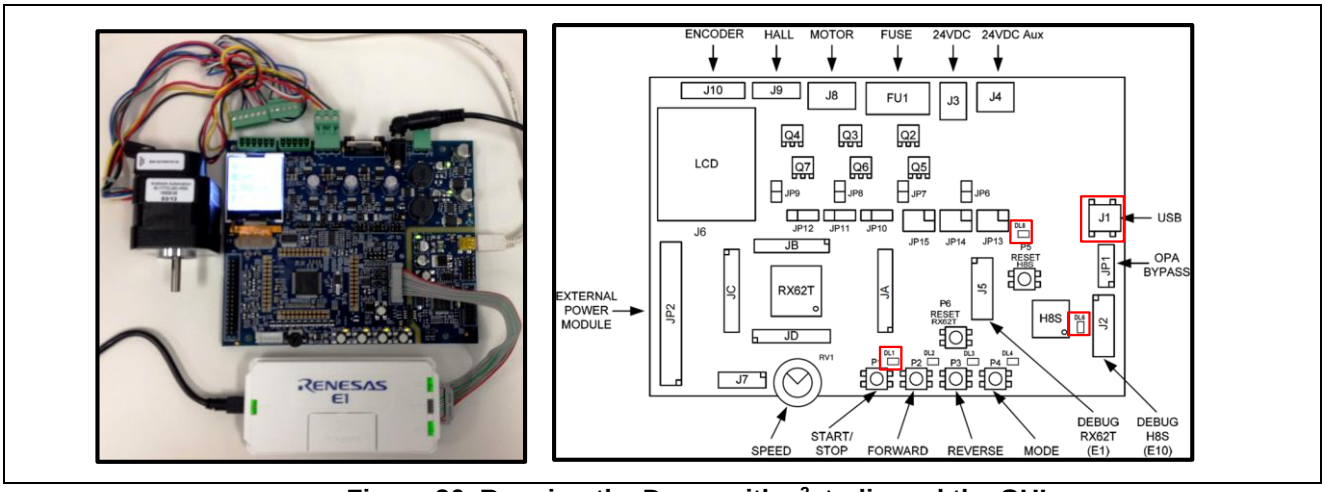


Figure 30 Running the Demo with e²studio and the GUI

Step 2. Start the GUI program by double clicking on the Motor Control Demo icon [ ] or select the "Motor Control Demo" program from the Windows taskbar "Start" in "All Programs" under the "Motor Control Demonstrator" folder.

The GUI program screen will launch and display as shown in Figure 31. For a serial port update, wait for a few seconds to get the "Connect" button highlighted and then proceed to Step 3.

Note: If the "Connect" button is not highlighted, the GUI couldn't find the correct USB device driver for COM port setting.

Step 3. Click the "Communication Settings" tab on the top left of the GUI seen in Figure 31 and select "Auto detect" under the serial port drop-down tab.

Step 4. Click <Connect>.

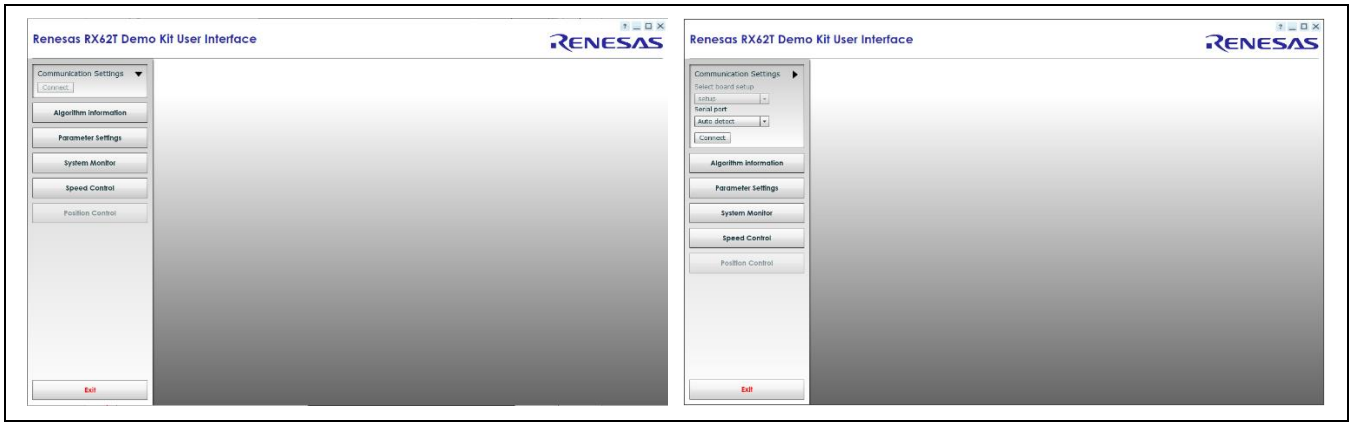


Figure 31 Connecting to the Motor Control Demo GUI 1 of 2

After successfully connecting with the target board, the "Communication Settings" area will change to a green color and the "Connect" button will change to "Disconnect." The LED **DL6** will blink while communicating between the target board and GUI. Figure 32 shows the GUI after a successful connection.

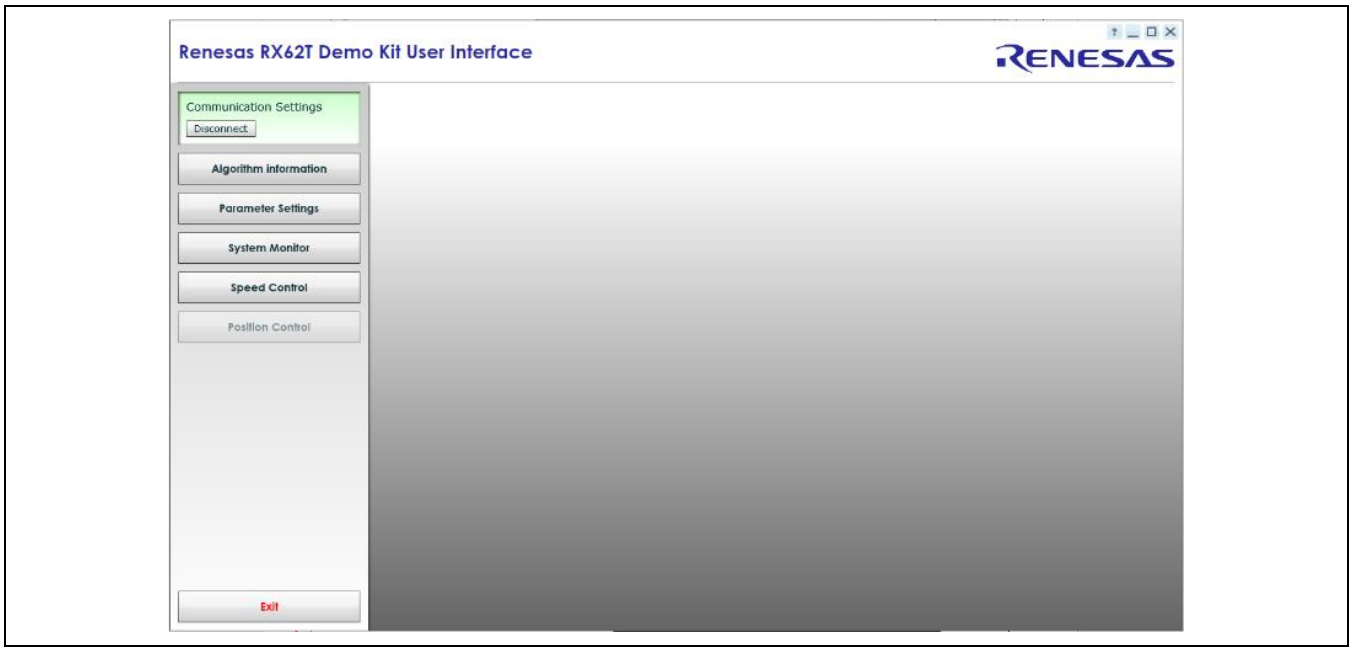


Figure 32 Connecting to the Motor Control Demo GUI 2 of 2

The GUI will detect the programmed algorithm. In this case the "Single-Shunt Sensorless Vector Control with External Amplifier" will be used. After connection, the "Speed Control" button is active while the "Position Control" button is grayed out. The user can check with the "Algorithm Information" message box which shows a valid algorithm. Clicking the "Verify Jumper Settings" button shows Table 4 in the GUI. Figure 33 shows the "Algorithm Information" dialog box. Follow the below procedure for using the GUI.

The LED **DL1** will be blinking continuously while running demo with no fault occurrence. If a fault occurs, the LED **DL2** will flash and **DL1** will remain illuminated without flashing. If a fault occurs, press **P6** (reset) and check if **DL1** begins to blink. If pressing **P6** does not fix the fault, disconnect and reconnect the E1 debugger and the Mini-USB and reprogram the board using the steps discussed in section 9.3.2.

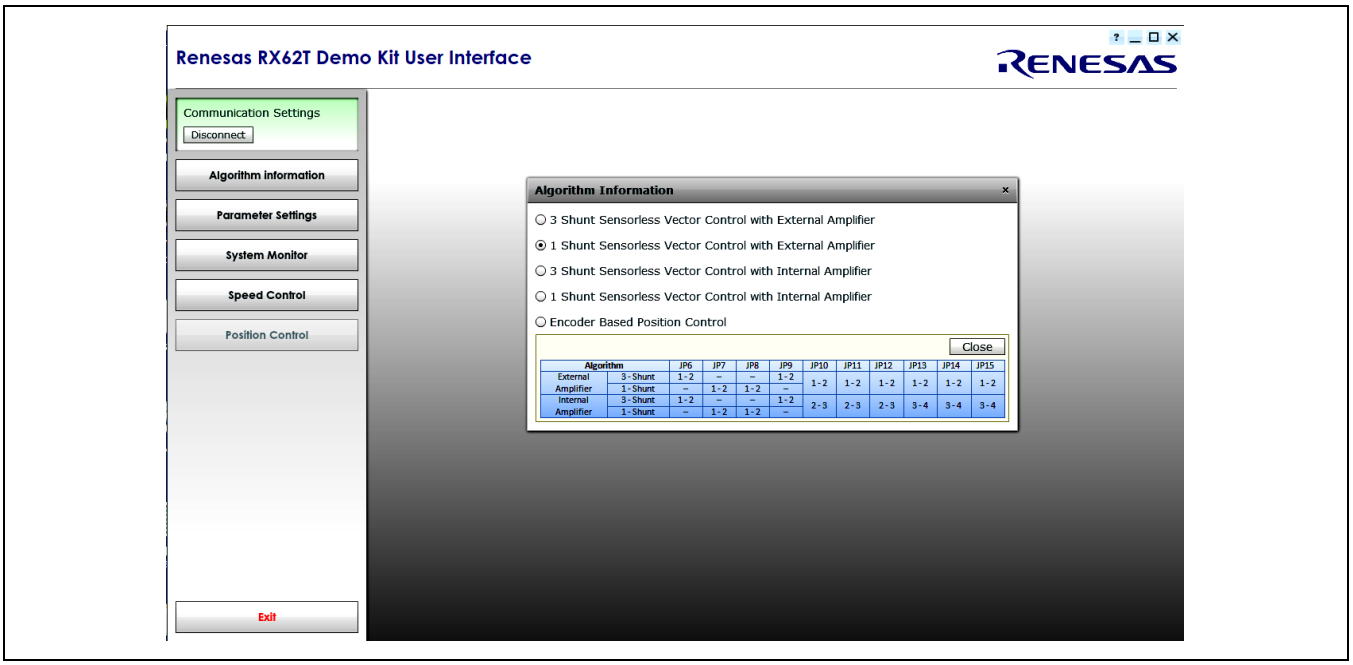


Figure 33 Algorithm Information in the GUI Application

Step 5. Click the <Speed Control> button.

Step 6. Set the speed arbitrarily by dragging the indicator needle to the right or left as shown in Figure 34. The speed can also be manually typed into the dialog box below the needle shown in Figure 34.

Note: The user needs to set the directional speed value from 2000 to 4000 rpm to run this demo.

The motor shaft should rotate with the setting speed. Returning the control needle to zero position stops the motor. By default, the demo sets parameter values for speed.

- Minimum speed 2000 rpm
- Maximum speed 4000 rpm
- Acceleration 5000 rpm/s
- Deceleration 5000 rpm/s
- Startup time 1000 ms

Figure 34 Setting the Speed to Turn the Motor in the GUI Application

Step 7. Click the "Parameter Settings" button.

The "Parameter Settings" feature can be used to manually adjust the preset variables using the GUI. Figure 35 shows the "Parameter Settings" dialog box.

Figure 35 Changing the Parameter Settings in the GUI Application

If standalone mode is used the target board will "Disconnect" and no longer be communicating with the GUI. The LED **DL6** will be on, but it will no longer be blinking. In order to reconnect from standalone mode, press **P6** (reset) and "Connect" to the GUI using the User Interface.

To terminate the GUI application, return the control needle to zero position, press the "Disconnect" button and then press the "Exit" button to close the application.

Appendix A - References

1. RX62T Group User's Manual: Hardware, R01UH0034EJ0110, April 20, 2011
2. DevCon 2010 Courses:
   - ID-620C, Complete Motor Control Integration with RX62T.
   - ID 623C, Understanding Sensorless Vector Control with Floating-Point Unit (FPU) Implementation.
3. Application Note of Sensorless Vector Control of Three-Phase PMSM Motors, REU05B0103-0100/Rev.1.00, March, 2009
4. Application Note of Mcrp05: Brushless AC Motor Reference Platform, REU05B0051-0100, Feb, 2009
5. Huangsheng Xu, Rohan Hubin, and Dave Cocca, "Sensorless Vector Control of PMSM Motor Using One Shunt Current Detection", IEEE-IAS 2008, Oct. 5-9, Edmonton, Alberta, Canada.
6. Huangsheng Xu, and Yashvant Jani, "Understanding Sensorless Vector Control for Brushless DC Motors", ESC-2008, Embedded System Silicon Valley Conference, April 15-17, San Jose, California, USA.

Website and Support

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

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

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Nov. 18, 2011.	—	First edition issued
2.00	Jan. 31, 2014.	—	Second edition issued
2.01	Jul. 30, 2014.	19	Demonstration Guide added

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the "Handling of MPU/MCU Products" and the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   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 (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable 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.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(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.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 LanGao Rd., Putuo District, Shanghai, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 3.0