User's Manual

E8a Emulator

Additional Document for User's Manual
R0E00008AKCE00EP4

Renesas Microcomputer Development Environment System
M16C Family / M16C/60 Series
Notes on Connecting the M16C/62P, M16C/6N4, M16C/6N5, M16C/6NK, M16C/6NM, M16C/6NL and M16C/6NN

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corporation without notice. Please review the latest information published by Renesas Electronics Corporation through various means, including the Renesas Electronics Corporation website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.4.00 Apr 2010

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. 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.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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.
5. When exporting the 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. You should not use Renesas Electronics products or the 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. 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. 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 categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. 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 an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “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.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. 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.
9. 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 system 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. 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.
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.

Contents

This user's manual is applicable to the E8a emulator software V.1.02 Release 00 or later.

Section 1 Inside the E8a Emulator User's Manual

The E8a emulator manual consists of two documents: the E8a User's Manual and the E8a Additional Document for User's Manual (this document). Be sure to read BOTH documents before using the E8a emulator.

(1) E8a Emulator User's Manual
The E8a Emulator User's Manual describes the hardware specifications and how to use the emulator debugger.

- E8a emulator hardware specifications
- Connecting the E8a emulator to the host computer or user system
- Operating the E8a emulator debugger
- Tutorial: From starting up the E8a emulator debugger to debugging

(2) E8a Additional Document for User's Manual
The E8a Additional Document for User's Manual describes content dependent on the MCUs and precautionary notes.

- MCU resources used by the E8a emulator
- Example of the E8a emulator connection or interface circuit necessary for designing hardware
- Notes on using the E8a emulator
- Setting the E8a emulator debugger during startup

Section 2 E8a Emulator Specifications

Table 2.1 shows the E8a emulator specifications for the M16C/62P and M16C/6N Groups.
This manual describes the M16C/6N4, M16C/6N5, M16C/6NK, M16C/6NM, M16C/6NL and M16C/6NN Groups as the M16C/6N Group.

Table 2.1 E8a Emulator Specifications for the M16C/62P and M16C/6N Groups

Target MCUs	M16C Family M16C/60 Series M16C/62P and M16C/6N Groups	
Available operating modes	Single-chip mode, Memory expansion mode * Microprocessor mode is not supported.	
Break functions	- Address match break, 8 points - PC break points (maximum 255 points) - Forced break	
Trace functions	None	
Flash memory programming function	Available	
User interface	Clock-synchronized serial (communication via P64/P65/P66/P67)	
MCU resources to be used	- ROM size: 2 KB (variable assigned address) - RAM size: 128 bytes (variable assigned address) - Stack 14 bytes - UART1 function and P64/P65/P66/P67 - Pins P50 and P55 - Address match interrupt	
Emulator power supply	Unnecessary (USB bus powered, power supplied from the PC)	
Interface with host machine	USB (USB 1.1, full speed) * Also connectable to host computers that support USB 2.0	
Power supply function	Can supply 3.3 V or 5.0 V to the user system (maximum 300 mA)	
Power voltages	M16C/62P	3.0 - 3.6 V, 4.5 - 5.5 V
	M16C/6N4 (Normal version) M16C/6N5 (Normal version) M16C/6NK (Normal version) M16C/6NL M16C/6NN	3.0 - 3.6 V, 4.5 - 5.5 V
	M16C/6N4 (T version, V version) M16C/6N5 (T version, V version) M16C/6NK (T version, V version) M16C/6NM (T version, V version)	4.5 - 5.5 V

Table 2.2 shows the operating environment of the E8a emulator.

Table 2.2 Operating Environment

Temperatures	Active	: 10°C to 35°C
	Inactive	: –10°C to 50°C
Humidity	Active	: 35% RH to 80% RH, no condensation
	Inactive	: 35% RH to 80% RH, no condensation
Vibrations	Active	: maximum 2.45 m/s^2
	Inactive	: maximum 4.9 m/s^2
	Transportation	: maximum 14.7 m/s^2
Ambient gases	No corrosive gases	

Section 3 Connecting the E8a Emulator to the User System

Before connecting the E8a emulator to the user system, a connector must be installed in the user system so a user system interface cable can be connected. When designing the user system, refer to Figure 4.1 “E8a Connecting Connector Pin Assignments”, and Figures 5.1 to 5.4 “Example of an E8a Connection”.
Before designing the user system, be sure to read the E8a Emulator User’s Manual and related device hardware manuals.

Table 3.1 shows the recommended connector for the emulator.

Table 3.1 Recommended Connector

	Type Number	Manufacturer	Specification
14-pin connector	2514-6002	3M Limited	14-pin straight type

Connect E8a connecting connector pins 2, 6, 10, 12 and 14 firmly to the GND on the user system board. These pins are used as an electric GND and monitor the connection of the user system connector. Note the pin assignments for the user system connector.

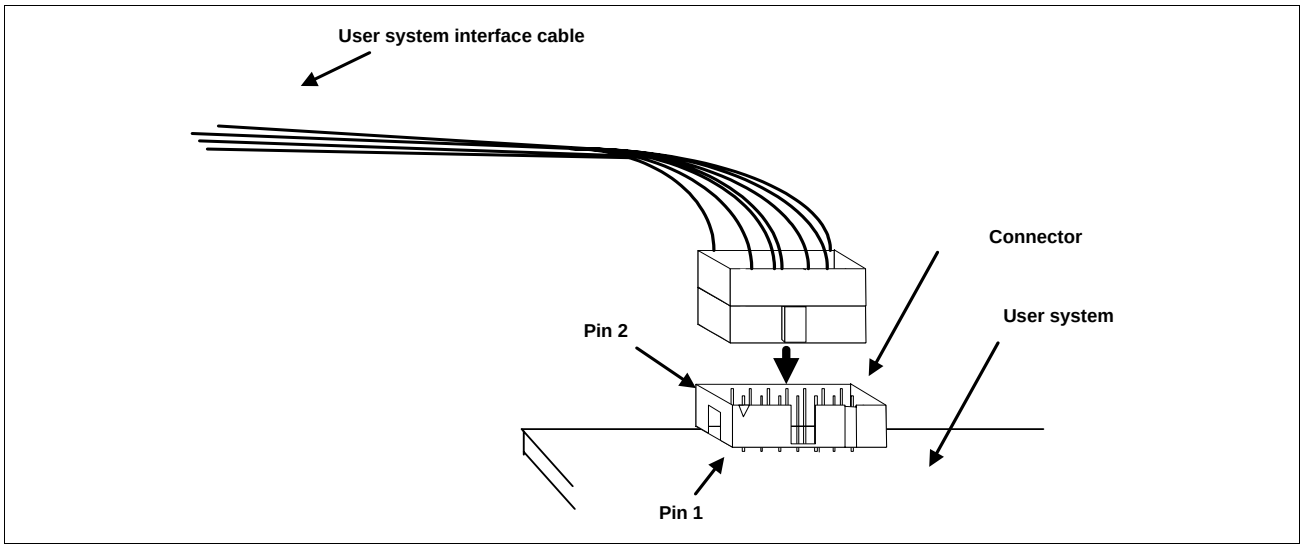


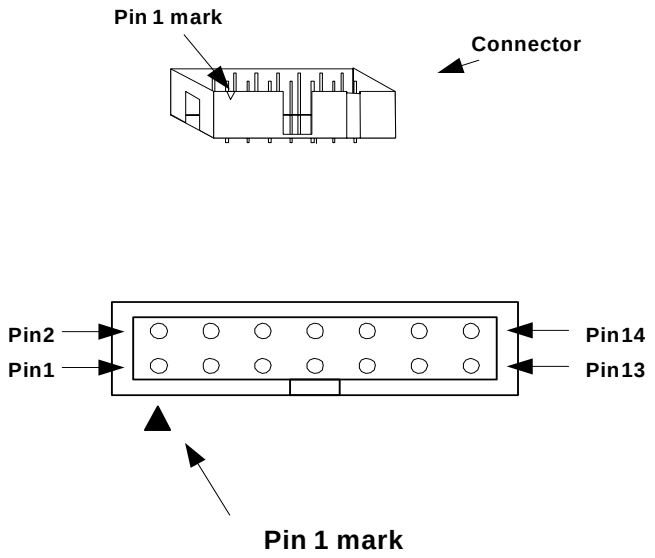
Figure 3.1 Connecting the User System Interface Cable with an E8a Connecting Connector

Notes:

1. Do not place any components within 3 mm area of the connector.
2. When using the E8a emulator as a programmer, connect it to the user system in the same way.

Section 4 E8a Connecting Connector Pin Assignments

Figure 4.1 shows the pin assignments for the E8a connecting connector.



Pin No.	M16C/62P and M16C/6N MCU signals
1	P65 (SCLK)
2	Vss
3	CNVss
4	P55 (EPM)
5	P67 (TxD)
6	Vss
7	P50 (CE)
8	Vcc
9	P64 (BUSY)
10	Vss
11	P66 (RxD)
12	Vss
13	$\overline{\text{RESET}}$
14	Vss

Figure 4.1 E8a Connecting Connector Pin Assignments

Note:
Pin 14 is used for checking the connection between the E8a and the user system, and is not directly connected to the Vss inside the E8a. Make sure pins 2, 6, 10, 12 and 14 are all connected to the Vss.

Section 5 Examples of E8a Connections

The following show connection examples. When using the emulator as a programmer, the connection specification between the E8a and the MCUs is the same as shown below.

(1) In single power supply and single-chip mode

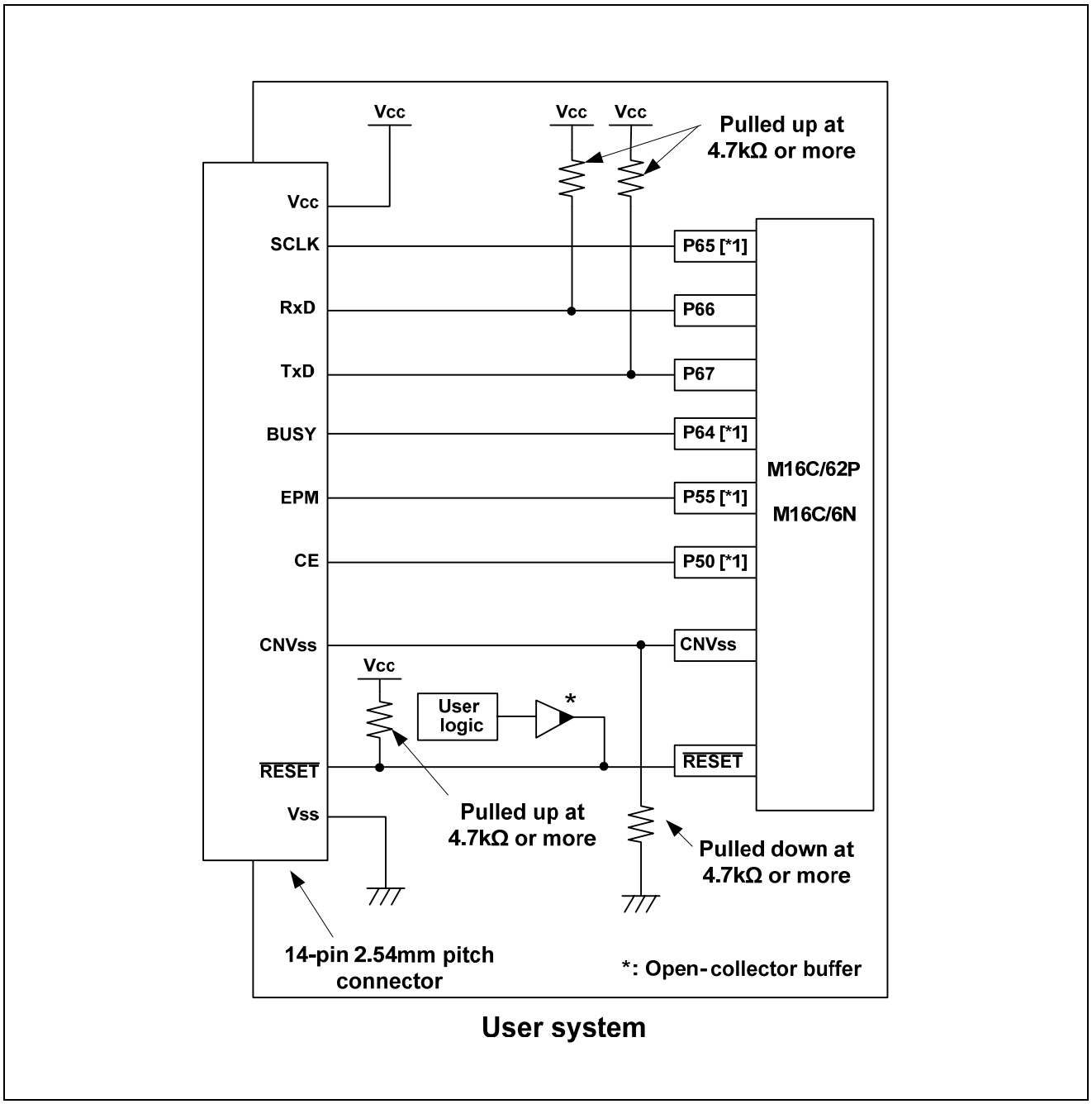


Figure 5.1 Example of an E8a Connection (Single Power Supply and Single-chip Mode)

Note:
1. For details on setting pins P50, P55, P64 and P65, refer to numbers 1 and 2 of “Points to Remember” on page 9.

(2) In single power supply and memory expansion mode

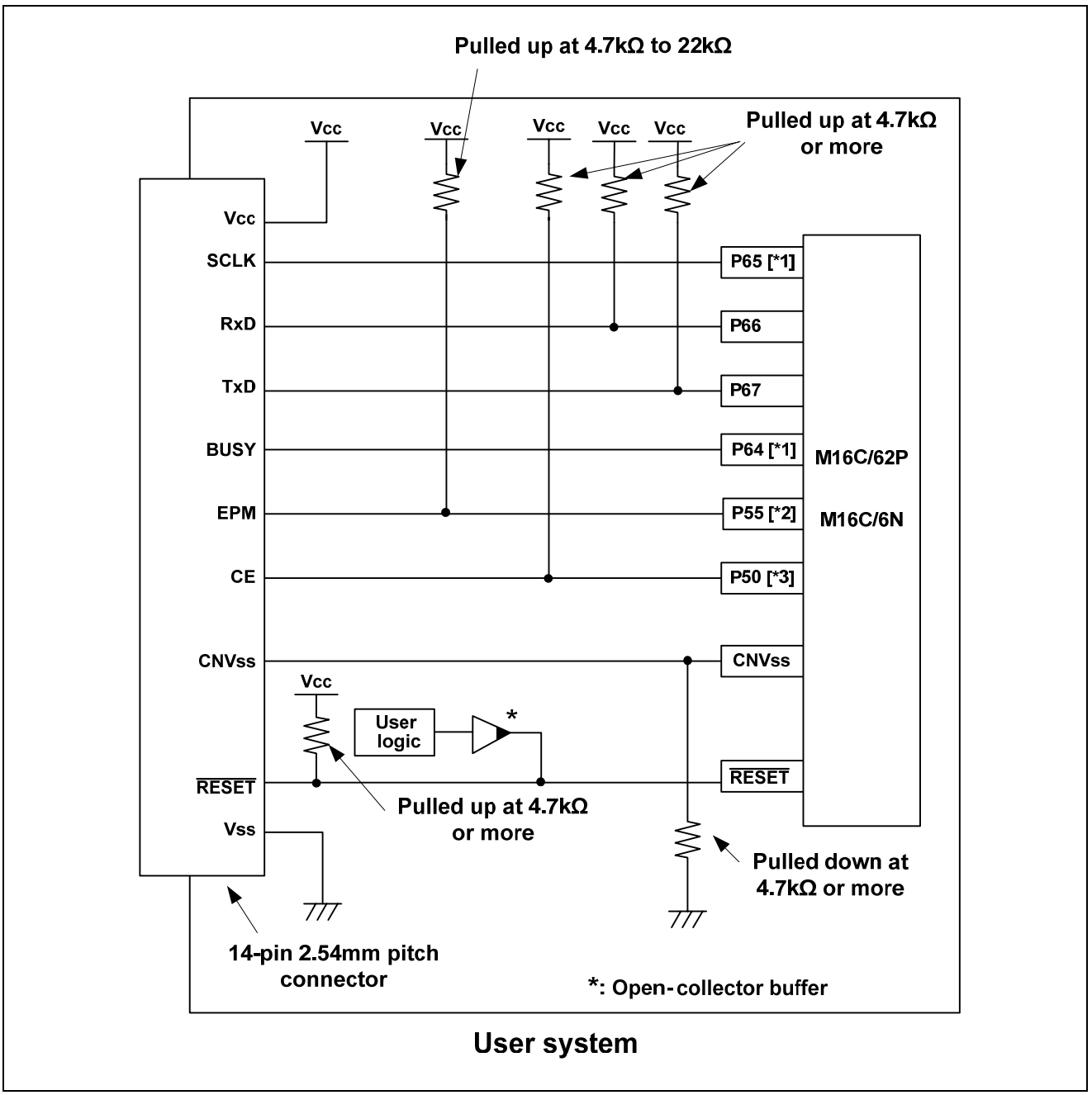


Figure 5.2 Example of an E8a Connection (Single Power Supply and Memory Expansion Mode)

Notes:

1. For details on setting pins P64 and P65 refer to number 1 of “Points to Remember” on page 9.
2. The $\overline{\text{HOLD}}$ signal cannot be used. Pull up P55 on the user system.
3. P50 is used as the WRL#/WR# pin. The E8a emulator outputs “H” to the CE pin when going to boot mode (resetting the MCU). In other cases, the CE pin is in a Hiz state. This prevents signal collision between the E8a emulator and the MCU. The WRL#/WR# pin does not affect the memory because the pin has a low active signal.

(3) In dual power supply and single-chip mode

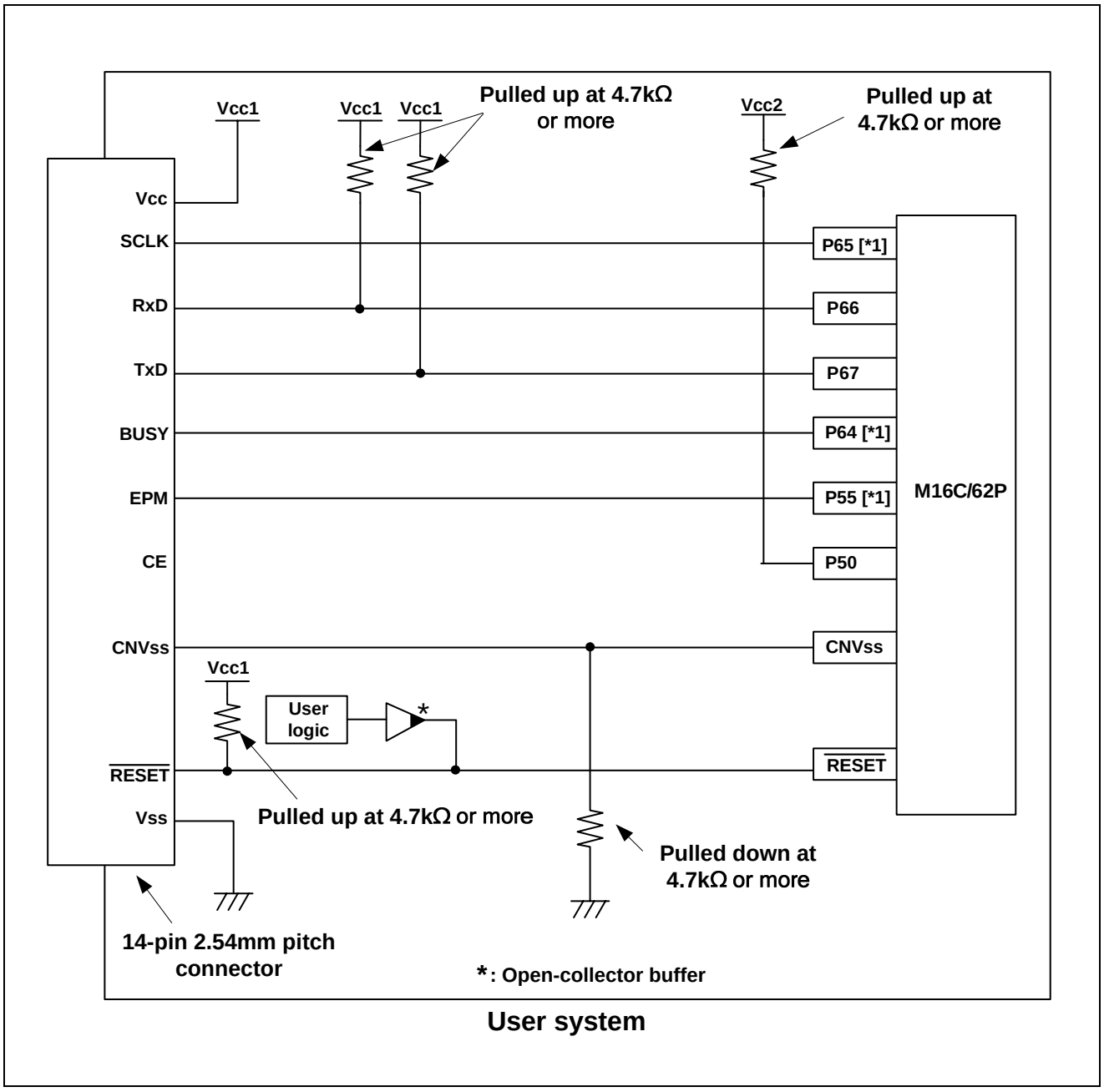


Figure 5.3 Example of an E8a Connection (Dual Power Supply and Single-chip Mode)

Note:
1. For details on setting pins P55, P64 and P65, refer to numbers 1 and 2 of “Points to Remember” on page 9.

(4) In dual power supply and memory expansion mode

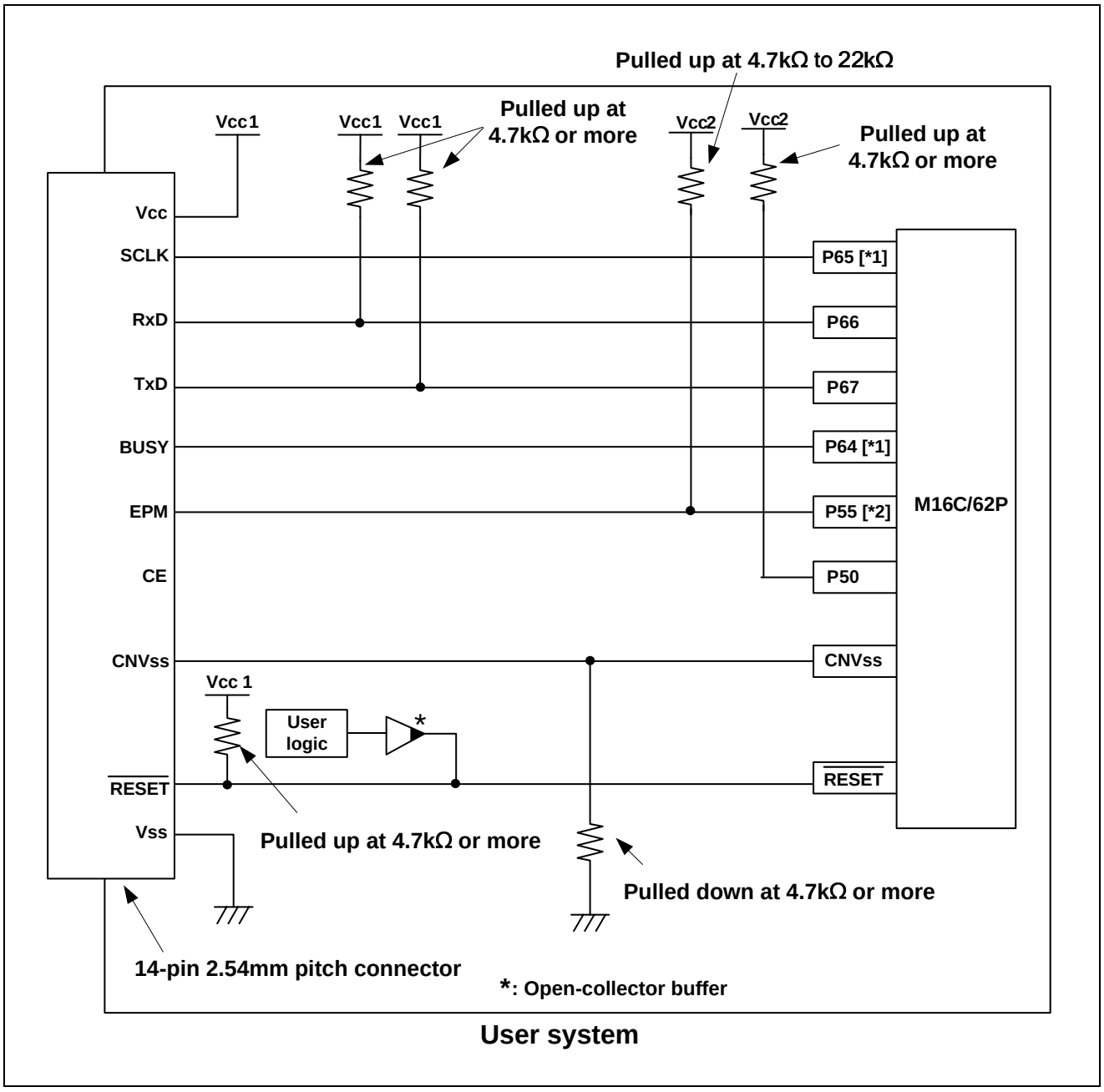


Figure 5.4 Example of an E8a Connection (Dual Power Supply and Memory Expansion Mode)

Notes:

1. For details on setting pins P64 and P65 refer to numbers 1 and 2 of “Points to Remember” on page 9.
2. The $\overline{\text{HOLD}}$ signal cannot be used. Pull up P55 on the user system.

Points to Remember

1. Pins P64, P65, P66 and P67 are used exclusively by the E8a emulator. Connect the E8a emulator to the MCU pins. Connect pins P66 and P67 to the E8a emulator after pulling up the MCU pins at the Vcc (Vcc1) level. For P64 and P65, pull up the pins at the Vcc (Vcc1) level or pull down them according to the MCU pin state after disconnecting the E8a emulator. P64 may be in a Hiz state while the E8a emulator is active. Therefore, set the pin resistance value so the voltage cannot be at the midpoint potential, depending on the voltage dividing of the resistance inside the E8a emulator (Figure 5.12).

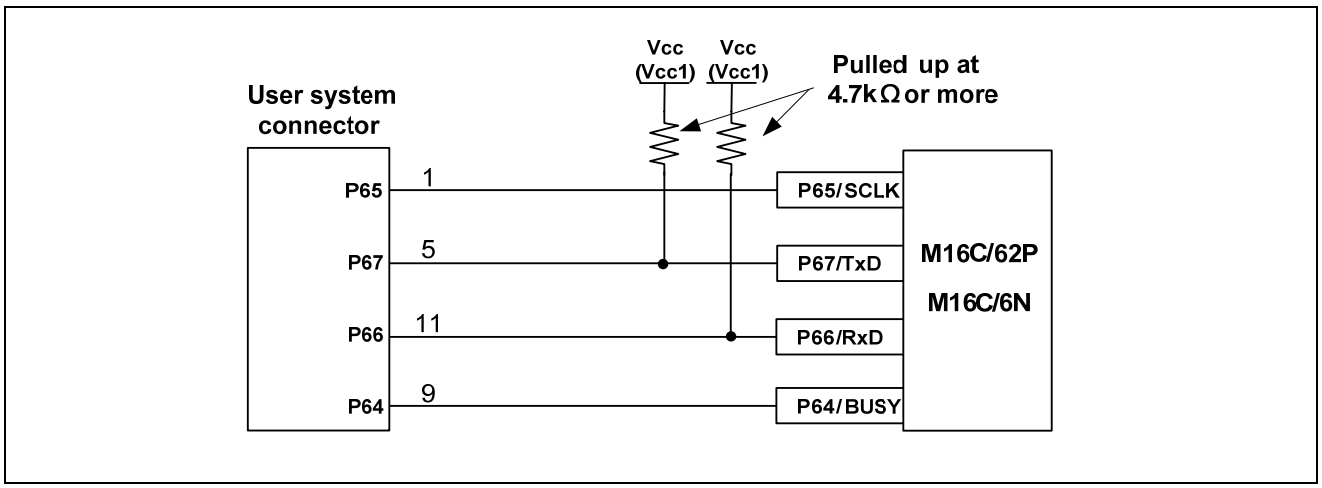


Figure 5.5 E8a Emulator and MCU Connection

2. The E8a emulator uses pins P50 and P55 for MCU control. Connect the E8a emulator to the MCU pins.

   (1) In single power supply and single-chip mode

   For P50 and P55, pull up the pins at the Vcc level or pull down them according to the MCU pin state after disconnecting the E8a emulator. P50 and P55 may be in a Hiz state while the E8a emulator is active. Therefore, set the pin resistance value so the voltage cannot be at the midpoint potential, depending on the voltage dividing of the resistance inside the E8a emulator (Figure 5.12).

Figure 5.6 Connection of E8a Emulator and Pins P50 and P55 (Single Power Supply and Single-chip Mode)

(2) Single power supply and memory expansion mode

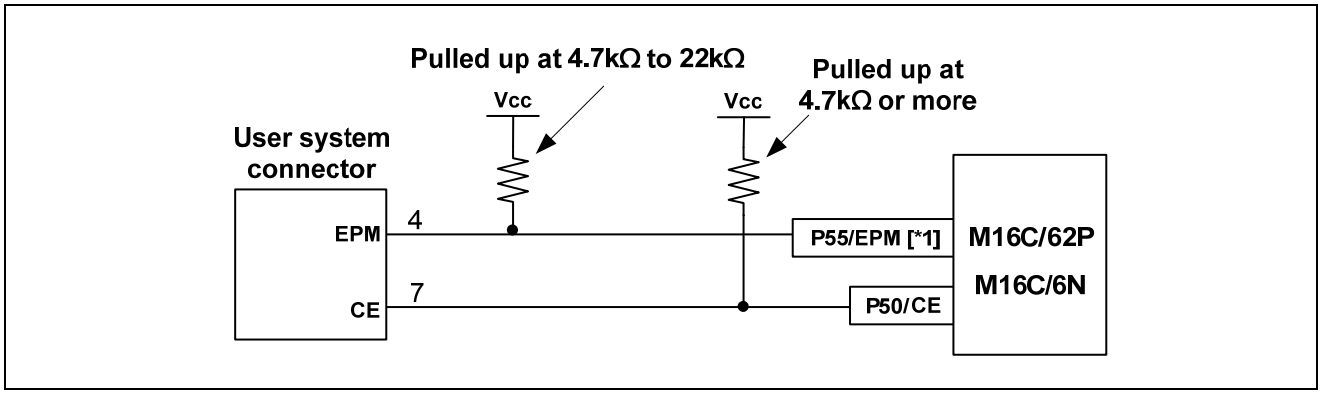


Figure 5.7 Connection of E8a Emulator and Pins P50 and P55 (Single Power Supply and Memory Expansion Mode)

Note:

1. The $\overline{\text{HOLD}}$ signal cannot be used. Pull up P55 at the Vcc level on the user system.

(3) In dual power supply and single-chip mode

Pull up P55 at the Vcc2 level or pull down it according to the MCU pin state after disconnecting the E8a emulator. P55 may be in a Hiz state while the E8a emulator is active. Therefore, set the pin resistance value so the voltage cannot be at the midpoint potential, depending on the voltage dividing of the resistance inside the E8a emulator (Figure 5.12).

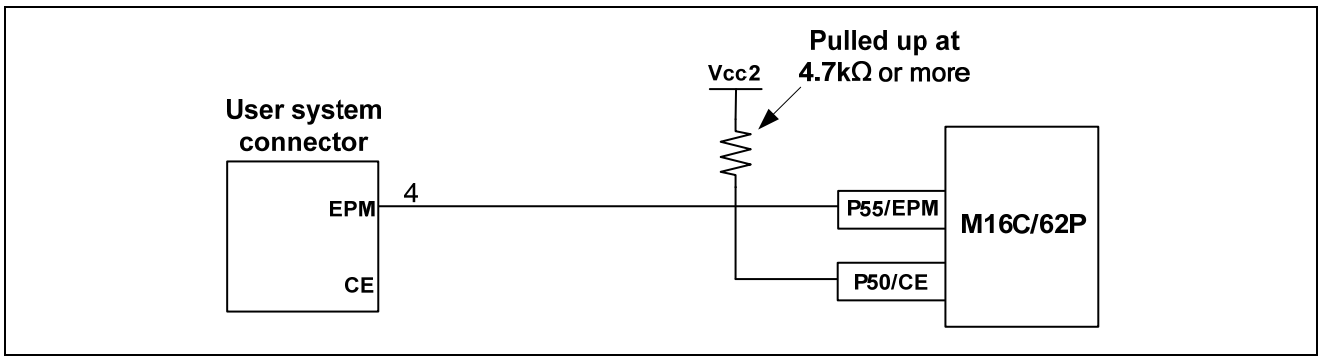


Figure 5.8 Connection of E8a Emulator and Pins P50 and P55 (Dual Power Supply and Single-chip Mode)

(4) In dual power supply and memory expansion mode

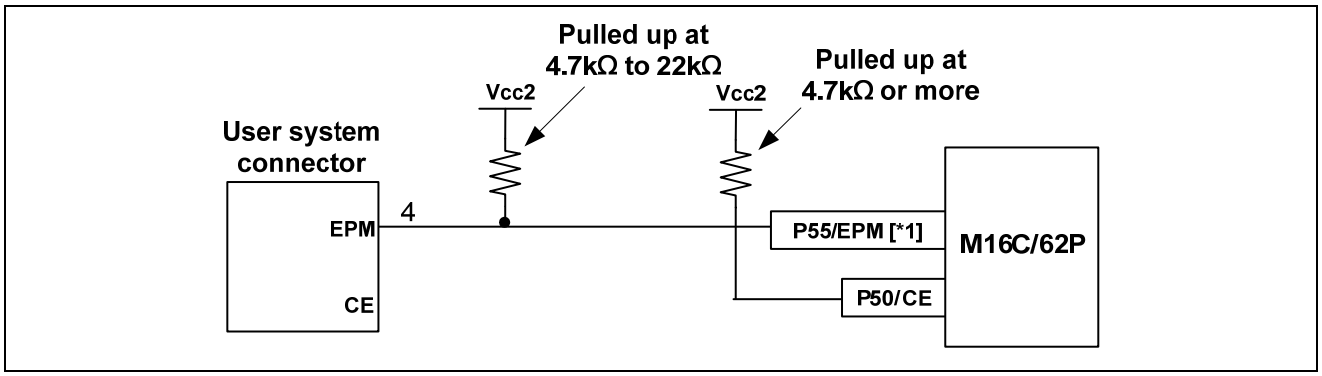


Figure 5.9 Connection of E8a Emulator and Pins P50 and P55 (Dual Power Supply and Memory Expansion Mode)

Note:

1. The $\overline{\text{HOLD}}$ signal cannot be used. Pull up P55 at the Vcc2 level on the user system.

3. The E8a emulator uses the CNVss pin for MCU control. Pull down the E8a emulator and MCU pins and connect the E8a emulator.

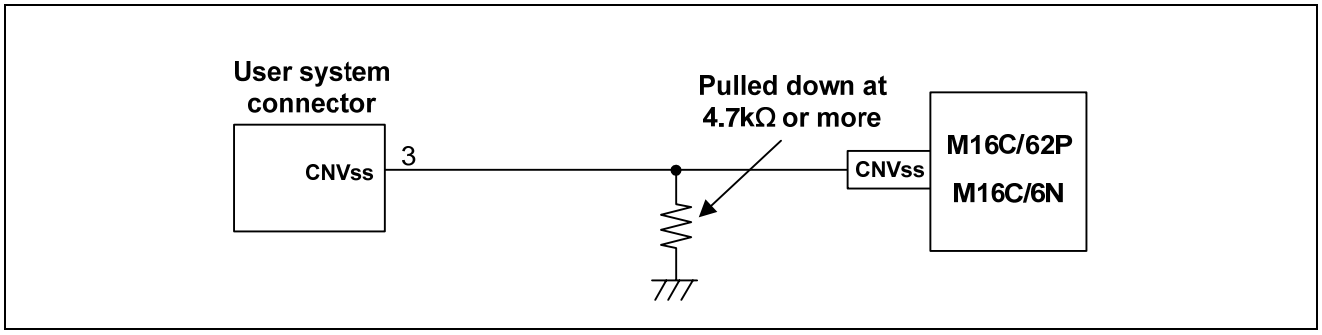


Figure 5.10 E8a Emulator and CNVss Pin Connection

4. The $\overline{\text{RESET}}$ pin is used by the E8a emulator. Therefore, use an open-collector output buffer or a CR reset circuit as the reset circuit for the user system. The recommended pull-up value is 4.7 kΩ or more. The MCU can be reset by outputting "L" from the E8a emulator. However, if the reset IC output is "H", the user system reset circuit cannot be set to "L". As such, the E8a emulator will not operate normally.

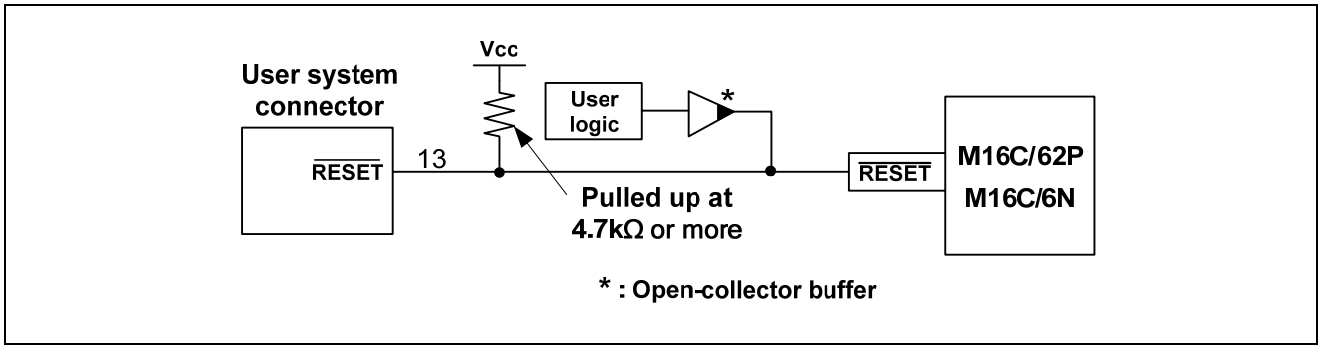


Figure 5.11 Example of a Reset Circuit

5. Connect Vss and Vcc to the Vss and Vcc (Vcc1) of the MCU, respectively.
6. The amount of voltage input to Vcc (Vcc1, Vcc2) must be within the specified range of the MCU.
7. If $\overline{\text{NMI}}$ interrupts are not used, make sure the $\overline{\text{NMI}}$ pin is pulled up to the Vcc (Vcc1) pin through a resistor.
8. Pin 14 is used for checking the connection between the E8a and the user system, and is not directly connected to the Vss inside the E8a. Make sure that pins 2, 6, 10, 12 and 14 are all connected to the Vss.

9. Figure 5.12 shows the interface circuit in the E8a emulator. Use this figure as a reference when determining the pull-up resistance value.

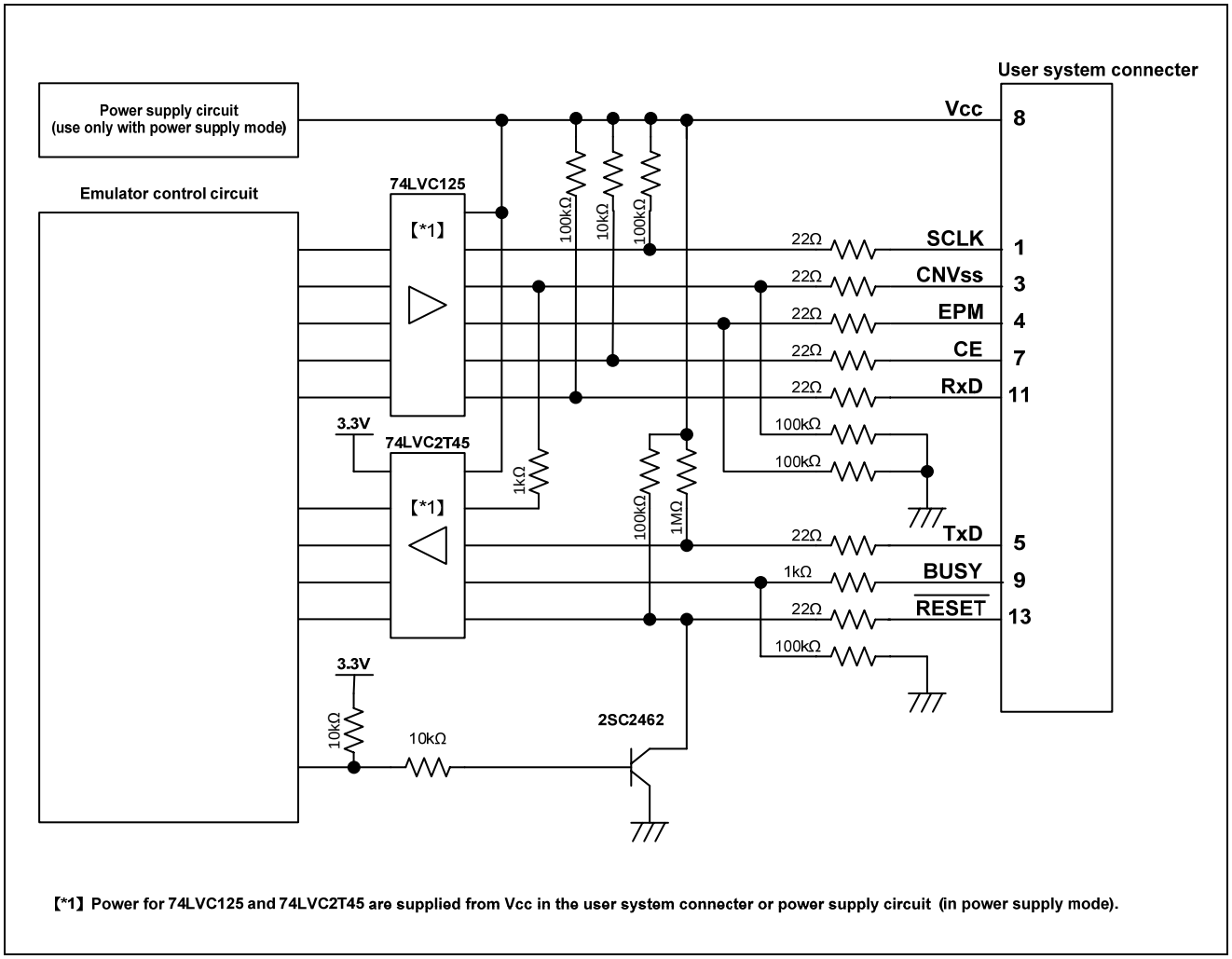


Figure 5.12 Interface Circuit Inside the E8a Emulator (For Reference)

Section 6 Notes on Using the E8a Emulator

1. Program area for the E8a emulator
   Table 6.1 lists the program areas allotted for the E8a emulator. Do not change this area allocation, otherwise the E8a emulator will not control the MCU. If settings were changed, disconnect the debugger and then reconnect it.

Table 6.1 Program Area for the E8a Emulator

Group	Part No.	ROM Size		RAM Size	Program Area for E8a Emulator		
		Programming Area	Data Area		Vector Area	ROM Area	RAM Area
M16C/62P	M30620FCP	128 KB	4 KB	10 KB	FFFE4h - FFFE7h, FFFE8h - FFFEBh, FFFECh - FFFEFh, FFFF4h - FFFF7h, FFFFCh - FFFFFh	2 KB of the programming area [*1]	128 bytes [*1]
	M30621FCP	128 KB		10 KB			
	M30622F8P	64 KB		4 KB			
	M30623F8P	64 KB		4 KB			
	M30624FGP	256 KB		20 KB			
	M30625FGP	256 KB		20 KB			
	M30626FHP	384 KB		31 KB			
	M30626FJP	512 KB		31 KB			
	M30627FHP	384 KB		31 KB			
	M30627FJP	512 KB		31 KB			
	M3062LFGP	256 KB		20 KB			
	M3062AFC	128 KB		10 KB			
	M3062CF8	64 KB		4 KB			
	M3062JFH	384 KB		31 KB			
M16C/6N4	M306N4FC	128 KB		5 KB			
	M306N4FG	256 KB		10 KB			
M16C/6N5	M306N5FC	128 KB		5 KB			
M16C/6NK	M306NKFH	384 KB		31 KB			
	M306NKFJ	512 KB		31 KB			
M16C/6NM	M306NMFH	384 KB		31 KB			
	M306NMFJ	512 KB		31 KB			
M16C/6NL	M306NLFH	384 KB		31 KB			
	M306NLFJ	512 KB		31 KB			
M16C/6NN	M306NNFH	384 KB		31 KB			
	M306NNFJ	512 KB		31 KB			

Note:
1. When starting the debugger, the [Emulator Setting] dialog box shown in Figure 6.1 is displayed. Specify the area which will not be used in the user system. The data area cannot be specified.

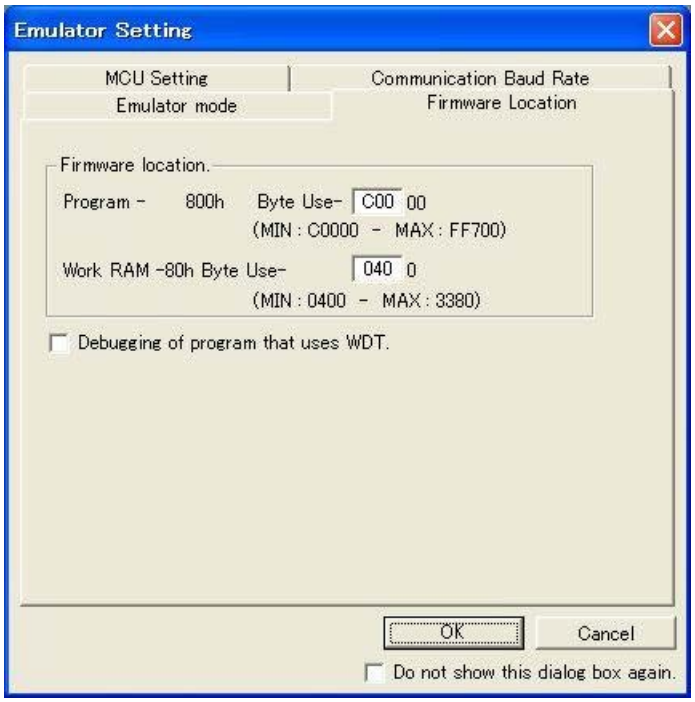


Figure 6.1 [Firmware Location] Tab of the [Emulator Setting] Dialog Box

2. When the system is launched, the E8a emulator initializes the general registers and some of the flag registers as shown in Table 6.2.

Table 6.2 E8a Emulator Register Initial Values

Status	Register	Initial Value
E8a Emulator Activation	PC	Reset vector value in the vector address table
	R0 to R3 (bank 0, 1)	0000h
	A0, A1 (bank 0, 1)	0000h
	FB (bank 0, 1)	0000h
	INTB	0000h
	USP	0000h
	ISP	Work RAM Address for the E8a emulator + 80h [*1]
	SB	0000h
	FLG	0000h

Note:
1. The Work RAM address for the E8a emulator is specified in the [Firmware Location] tab of the [Emulator Setting] dialog box.

3. The E8a emulator controls the MCUs by using the P50, P55, P64, P65, P66, P67, $\overline{\text{RESET}}$ and CNVss pins.

4. The E8a emulator uses up to 14 bytes of the stack pointer (ISP) during a user program break. Therefore, set aside 14 bytes for the stack area.

5. SFRs used by the E8a emulator program

As the SFRs listed in Table 6.3 are used by the E8a emulator program, do not change any of these values. If these values are changed, the E8a emulator cannot control the MCU. Note that UART1 transmit interrupt control register S1TIC and UART1 receive interrupt control register S1RIC always read out values used by the emulator. These registers are not initialized by selecting [Debug] -> [Reset CPU] or by using the RESET command. If register contents are referred to, a value that has been set in the E8a emulator program will be read out.

Table 6.3 SFRs Used by the E8a Emulator Program

Address	Register	Symbol	Bit	Notes on Using the E8a Emulator
0009h	Address match interrupt enable register	AIER	All bits	[*1]
0010h - 0012h	Address match interrupt register 0	RMAD0	All bits	[*1]
0014h - 0016h	Address match interrupt register 1	RMAD1	All bits	[*1]
01B8h - 01BAh	Address match interrupt register 2	RMAD2	All bits	[*1]
01BBh	Address match interrupt enable register 2	AIER2	All bits	[*1]
01BCh - 01BEh	Address match interrupt register 3	RMAD3	All bits	[*1]
03A8h	UART1 transmit/receive mode register	U1MR	All bits	[*1]
03AAh, 03ABh	UART1 transmit buffer register	U1TB	All bits	[*1]
03ACh	UART1 transmit/receive control register 0	U1C0	All bits	[*1]
03ADh	UART1 transmit/receive control register 1	U1C1	All bits	[*1]
03AEh, 03AFh	UART1 receive buffer register	U1RB	All bits	[*1]
03B0h	UART transmit/receive control register 2	UCON	Bits 1, 3, 4, 5 and 6	[*2]
03ECh	Port P6 register	P6	Bits 4, 5, 6 and 7	[*2]
03EEh	Port P6 direction register	PD6	Bits 4, 5, 6 and 7	[*2]

Notes:
1. Do not change this register value.
2. Do not change the value of the bits listed in the column to the left. When operating this register, make changes using the bit operation instructions to avoid changing the bit values.

6. Interrupts used by the E8a emulator program

The BRK instruction interrupt, address match interrupt, single-step interrupt and DBC interrupt are used by the E8a emulator program. Therefore, make sure the user program does not use any of these interrupts. The E8a emulator changes these interrupt vector values to the values to be used by the emulator. No problems occur if the interrupt vector values are written in the user program.

7. Debugging using the watchdog timer

When debugging the user program using the watchdog timer, click the [Debugging of program that uses WDT.] check box in the [Firmware Location] tab of the [Emulator Setting] dialog box. By clicking this box, the E8a emulator program refreshes the watchdog timer during program operation. If memory access is executed through memory reference or modification, the watchdog timer will be refreshed by the E8a emulator program.

When using the actual MCU, the watchdog timer starts operating by writing to the watchdog timer start register. However, when using this emulator, the watchdog timer starts after initiating the user program because the E8a emulator program refreshes the watchdog timer even if a user program halts. Note that this timing will differ from the actual operational timing.

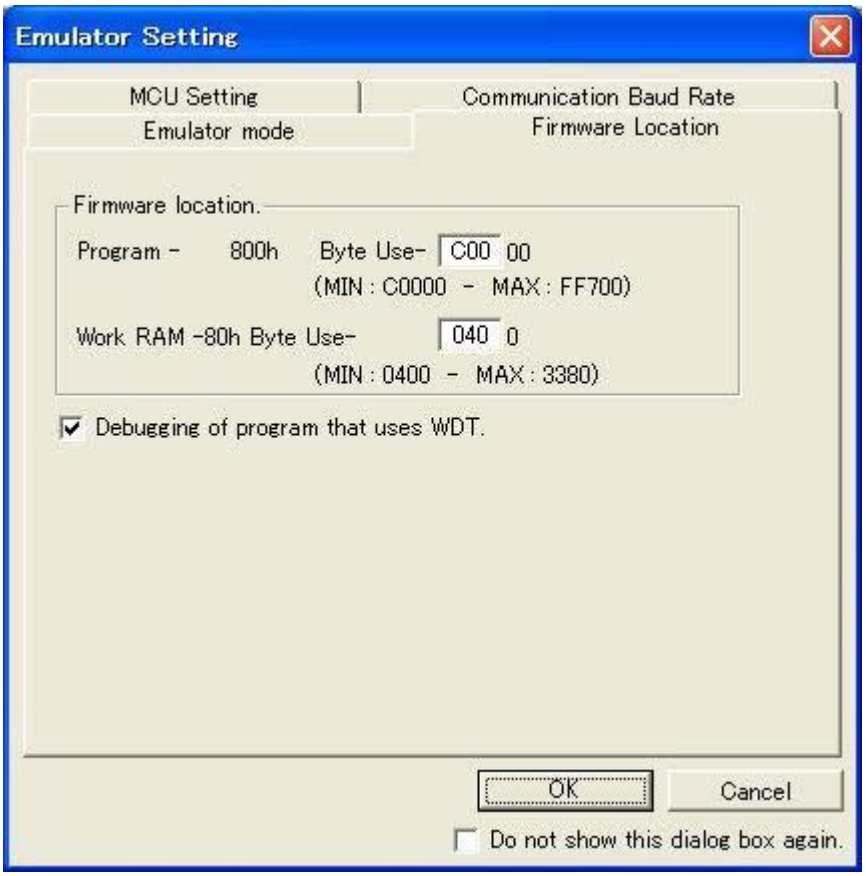


Figure 6.2 [Firmware Location] Tab of the [Emulator Setting] Dialog Box

8. Flash memory ID code

This MCU function prevents the Flash memory from being read out by anyone other than the user. The ID code in Table 6.4 written to the flash memory of the MCU must match the ID code displayed in the Figure 6.3 [ID Code verification] dialog box at debugger startup, otherwise the debugger cannot be launched. Note that when the ID code is FFh, FFh, FFh, FFh, FFh, FFh, FFh, the ID code is regarded as undefined. In this case, the ID code is automatically authenticated and the [ID Code verification] dialog box is not displayed.
In 'Program Flash' mode, the contents of the user program are input into the ID code area. When debugging in other modes, FFh, FFh, FFh, FFh, FFh, FFh, FFh is written into the ID code area regardless of the contents of the downloaded user program.

Table 6.4 ID Code Storage Area of M16C/62P and M16C/6N

Address	Description
FFFDFh	First byte of ID code
FFFE3h	Second byte of ID code
FFFEBh	Third byte of ID code
FFFEFh	Fourth byte of ID code
FFFF3h	Fifth byte of ID code
FFFF7h	Sixth byte of ID code
FFFFBh	Seventh byte of ID code

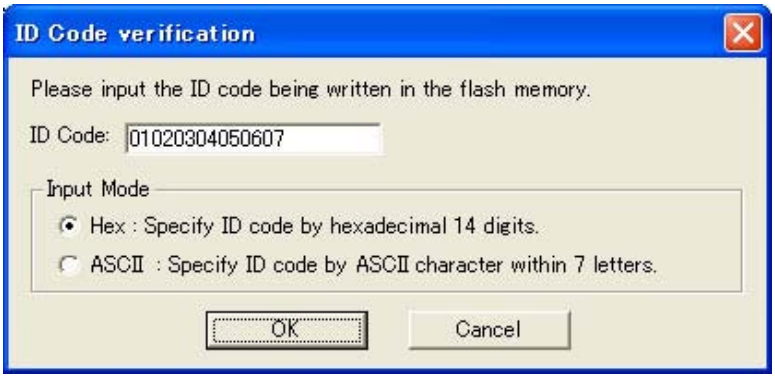


Figure 6.3 [ID Code verification] Dialog Box

Note on 'Program Flash' mode:

When the ID code is specified by the -ID option of the lmc30, download the MOT file or HEX file. When the X30 file is downloaded, the ID code is not valid. When downloading the X30 file, specify the ID code using an assembler directive command such as ".BYTE". The file to which the ID code specified by the assembler directive command ".ID" is output varies depending on the version of the assembler. For details, refer to the Assembler User's Manual.

9. Reset

The reset vector is used by the E8a emulator program. If the MCU is reset while executing the user program, control is transferred to the E8a emulator program and the user program is forced to stop. Do not use the hardware reset 2, software reset, watchdog timer reset and oscillation stop detection reset, otherwise the E8a emulator will not operate normally.

10. Memory access during emulation execution

    When referring to or modifying the memory contents, the user program is temporarily halted. For this reason, a real-time emulation cannot be performed. When a real-time emulation is necessary during a program execution, disable the automatic update in the watch window or fix the display in the memory window before running the program so that memory accesses do not occur during an execution.

11. Setting of address match break during user program execution

    When adding or cancelling the address match break, the user program is temporarily halted. For this reason, a real-time emulation cannot be performed.

12. When the E8a emulator does not supply power to the user system, it consumes the power voltage of the user system from several mA to more than 10 mA. This is because the user power supply drives 74LVC125, 74LVC1T45 and 74LVC2T45 to make the communication signal level match the user system power supply voltage.

13. When debugging, the Flash memory is frequently rewritten by the E8a emulator. Therefore, do not use an MCU that has been used for debugging in products. Also, as the E8a emulator program is written to the MCU while debugging, do not save the contents of the MCU Flash memory which were used for debugging nor use them as the ROM data for products.

14. NMI interrupt

    If NMI interrupts are used, be sure to take the necessary precautions before executing the user program like disabling the automatic update in the watch window or fix the display in the memory window before running the program so that memory accesses do not occur during an execution. If an NMI interrupt occurs while the user program halts or when memory contents are referenced or modified during user program execution, the E8a emulator cannot control the MCU.

15. Reserved area

    The addresses not specified in the Hardware Manual for M16C/62P and M16C/6N Groups are reserved area. Do not change the contents. Otherwise, the E8a emulator cannot control the MCU.

16. Debugging in stop mode or wait mode

    When debugging in stop mode or wait mode, do not operate windows until the program stops at the breakpoint by setting the breakpoint at the line of the program which will be executed after the stop mode or wait mode is cancelled. In addition, disable the automatic update in the watch window or fix the display in the memory window before running the program so memory accesses do not occur during an execution.
    When the program is forcibly stopped or when the memory is referred to or modified in stop mode or wait mode, these mode will be cancelled.

17. Peripheral I/Os during a halt

    During a user program halt, interrupts are not accepted although peripheral I/Os continue to run. For example, a timer interrupt is not accepted although the timer continues to count when a user program is stopped by a break after the timer started.

18. Exceptional step execution

a) Software interrupt instruction

Step execution cannot be performed in the internal processing of instructions (undefined, overflow, BRK and INT) which generate a software interrupt continuously in the program.

Example: INT instruction

```
        NOP
        NOP
        INT  #3          ↘ Passes through if the STEP execution is carried out.
        NOP
        JMP MAIN
INT_3:
    NOP          ←——  Program should be stopped at this address.
    NOP
    NOP
    REIT
```

b) INT instruction

To debug the user program with the INT instruction, set a PC break for the internal processing of the INT instruction and execute the program with the GO command.

Example:

```
        NOP
        INT  #3
        NOP                  ↓ Execute using GO command.
        JMP  MAIN
INT_3:
NOP  Break  ←——
NOP
REIT
```

19. “Go to cursor” function

The “Go to cursor” function is actualized using an address match break. Therefore, when you execute the “Go to cursor” command, all the address match breaks you set become invalid, while all the PC breaks remain valid.

20. Note on PC break point

When downloading a user program after modifying it, the set address of PC break may not be corrected normally depending on the modification. Therefore, break points other than the set PC breaks may shift. After downloading a user program, check the setting of PC breaks in the event point window and reset it. If a low-speed clock such as the sub clock is used as the operation clock of the MCU, setting or canceling PC breaks may take time. Use address match breaks as the first choice.

21. Note on debugging in CPU rewrite mode

When debugging in CPU rewrite mode, do not rewrite in CPU block 0 area (addresses FF000h - FFFFFh) and block containing the E8a emulator program. If these areas are rewritten, the E8a emulator will not control the MCU. Do not halt the user program while setting up the CPU rewrite mode and releasing it. If halted, the E8a emulator may not control the MCU. In addition, disable the automatic update in the watch window or fix the display in the memory window before running the program so memory accesses do not occur during an execution.

To check the data after executing the CPU rewrite mode, halt the program after releasing the CPU rewrite mode and refer to the memory window, etc.

When rewriting the Flash memory in the program area, select Menu -> [Setup] -> [Emulator] -> [System...] to open the [Configuration] dialog box in the High-performance Embedded Workshop. In this dialog box, change the [Flash memory synchronization] setting to [Flash memory to PC] and set the debugger cache to OFF. In this setting, the Flash memory is read whenever a break occurs, which takes some time. Use it with the [Disable] setting except when debugging in CPU rewrite mode.

22. Note on lock bits of Flash memory

When starting up in the [Erase Flash and Connect] mode or [Program Flash] mode, lock bits in all the blocks of the Flash memory will be unlocked. Note that the lock bits of the downloaded blocks will be unlocked after downloading the user program.

23. Notes on rewriting Flash memory

Do not reset the MCU when rewriting the Flash memory.

The Flash memory is rewritten when the “Flash memory write end” is displayed in the output window of the High-performance Embedded Workshop. If the MCU is reset when rewriting the Flash memory, the user program or the E8a emulator program may be disrupted.

Flash memory rewrite occurs:

- When downloading the user program
- After setting PC breaks in the Flash memory and executing the user program
- After canceling PC breaks in the Flash memory and executing the user program
- After rewriting the value of the Flash memory in the memory window and executing the user program

24. Notes on the E8a emulator power supply

When writing a program with the E8a emulator for mass production processes, the program requires reliability, so do not use the E8a emulator power supply function. Supply power separately to the user system according to the allowable voltage for MCU writing. Voltage supplied from the E8a emulator depends on the quality of the USB power supply of the PC, and as such, precision is not guaranteed. Note that when debugging the system which operates the MCU with a dual power supply, power cannot be supplied from the E8a.

25. Low power consumption mode

When debugging in low power consumption mode, do not operate windows until the program stops at the breakpoint by setting the breakpoint at the line of the program which will be executed after the low power consumption mode is cancelled.

26. DMAC during a user program halt

When the user program is halted or when the memory is referred to or modified during user program execution, DMA transfer is disabled. In such cases, the E8a emulator sets the registers below as following. Therefore, if you refer to the registers below in the memory window, etc., it shows that DMA is disabled.

- DMA0 Control Register (DM0CON)
  DMA enable bit (bit 3) 0: DMA disabled
- DMA1 Control Register (DM1CON)
  DMA enable bit (bit 3) 0: DMA disabled
- Interrupt Control Register
  Interrupt request bit (bit 3) 0: Interrupt not requested [*1]

Do not enable DMA transfer from the memory window, etc., but enable it in the user program.

Note

[*1] When restarting the user program, though the E8a emulator sets back the value of a DMA enable bit to the previous value that was set before the program stops, the interrupt request bit remains 0.

Section 7 Debugger Setting

1. [Emulator Setting] dialog box

The [Emulator Setting] dialog box is provided for setting items that need to be set when the debugger is launched. The contents set from this dialog box (excluding [Power Supply] group box items) also become valid the next time the debugger is launched. When launching the debugger for the first time after creating a new project work space, the [Emulator Setting] dialog box is displayed with the Wizard.

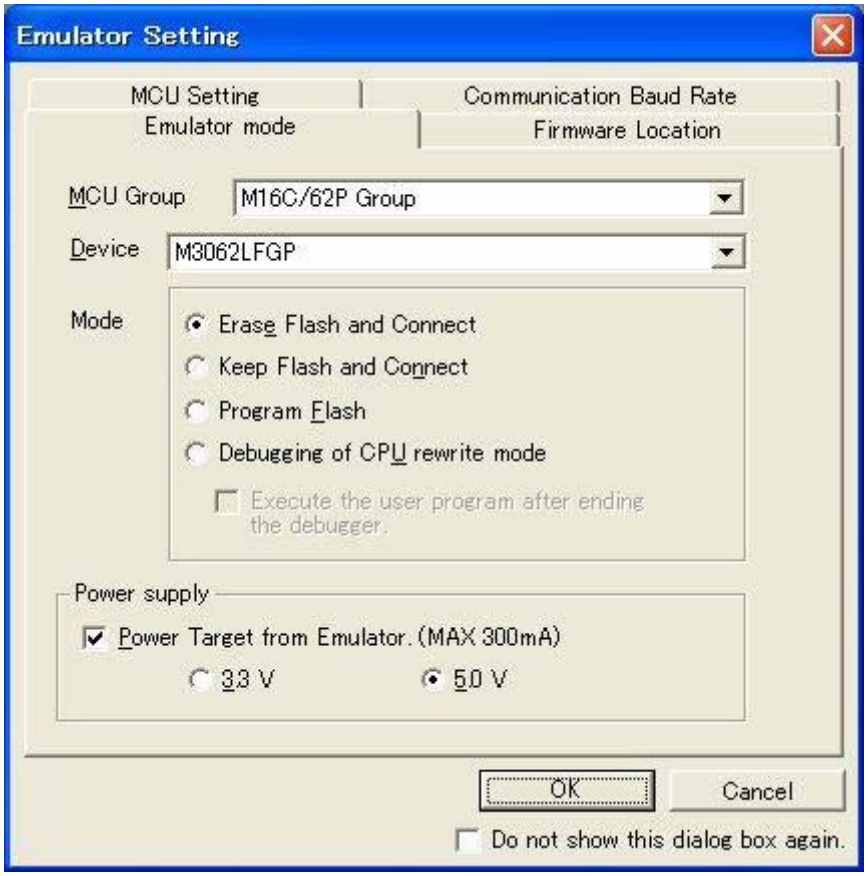


Figure 7.1 [Emulator Setting] Dialog Box

If you check “Do not show this dialog box again.” at the bottom of the [Emulator Setting] dialog box, the [Emulator Setting] dialog box will not be displayed the next time the debugger is launched. You can open the [Emulator Setting] dialog box using one of the following methods:

- After the debugger is launched, select Menu -> [Setup] -> [Emulator] -> [Emulator Setting...].
- Hold down the Ctrl key while launching the debugger.

When “Do not show this dialog box again.” is checked, the E8a does not supply power to the user system.

2. [Emulator mode] tab

Device selection, mode specification and power supply setting are made from the [Emulator mode] tab of the [Emulator Setting] dialog box.

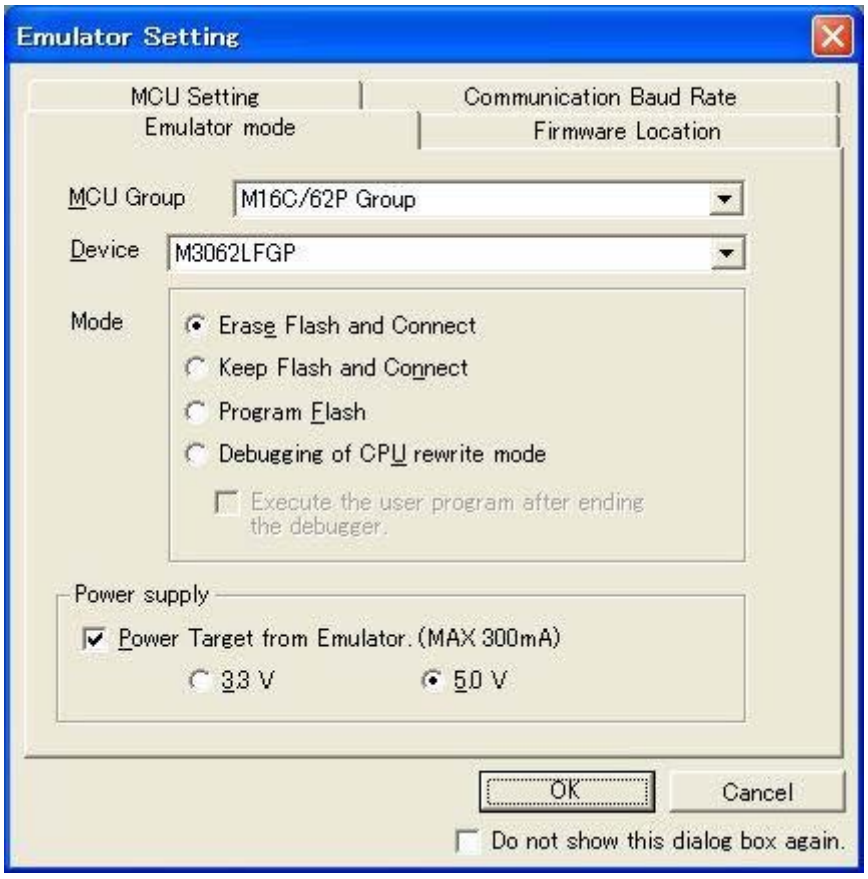


Figure 7.2 [Emulator mode] Tab

[MCU Group]

Select the name of the MCU group to be used from the [MCU Group] drop-down list.

[Device]

Select the type of MCU to be used from the [Device] drop-down list.

[Mode]

- Erase Flash and Connect
  When starting the debugger, the E8a emulator erases the Flash memory data for the MCUs and simultaneously writes the E8a emulator program.
- Keep Flash and Connect
  When launching the debugger, the E8a emulator retains the Flash memory data for the MCUs. Note that the area for the E8a emulator program and the vector area used by the E8a emulator will change.
- Program Flash
  The E8a emulator starts as a simple programmer. When downloaded, the E8a writes only the user program (E8a emulator program is not written). Therefore, the program cannot be debugged in this mode.
- Debugging of CPU rewrite mode
  Select this setting when debugging the program which rewrites the CPU. In this mode, the following debug operation which rewrites the Flash memory cannot be executed.

  - Setting the PC break points
  - Changing the memory contents in the Flash memory area

  In this mode, when starting the debugger, the E8a emulator erases the Flash memory data for the MCUs and simultaneously writes the E8a emulator program.

  When [Execute the user program after ending the debugger.] is selected, with the E8a emulator connected to the user system, the user program is executed at the same time the debugger is terminated. This check box setting is available only when the [Program Flash] mode is selected.

[Power supply]

When supplying power to the user system from the E8a, click the [Power Target from Emulator. (MAX 300mA)] check box. Note that when debugging the system which operates the MCU with a dual power supply, power cannot be supplied from the E8a.

3. [Firmware Location] tab

For details, see “1. Program area for the E8a emulator” and “7. Debugging using the watchdog timer” in “Section 6. Notes on Using the E8a Emulator”.

4. [MCU Setting] Tab

In the [MCU Setting] tab, set the operating condition of the MCU used in the user system.

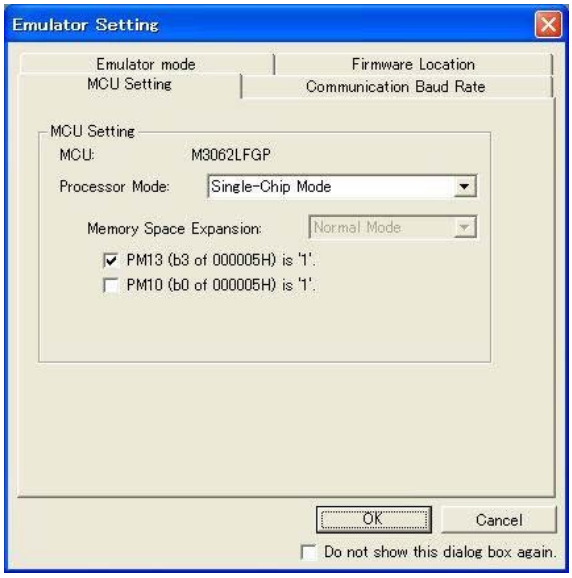


Figure 7.3 [MCU Setting] Tab

Specify processor mode

Specify the processor mode according to the user system. One of the following can be specified:

- Single-Chip Mode
- Memory Expansion Mode

Memory Expansion Mode

When Memory Expansion Mode is selected, specify whether the memory space expansion function will be used. When using the memory space expansion function, select “4 MB Mode”, and when not using, select “Normal Mode”.

PM13 (b3 of 000005H) is ‘1’

Specify whether PM13 (third bit of processor mode register 1) is set. When using the user program with PM13 set to “1”, check this option.

PM10 (b0 of 000005H) is ‘1’

Specify whether PM10 (zero bit of processor mode register 1) is set. When using the user program with PM10 set to “1”, check this option.

Notes:

The following describes precautions to be taken when using the emulator in memory expansion mode:

- When the external area cannot be rewritten via normal memory access, software breaks cannot be used in that area.
- No address match break can be used in the external memory space.
- The “Go to Cursor” function cannot be used in the external memory space. If this function is used in the external memory space, the program will be in a state of execution.
- To access the memory space expansion area in the download, the editor window (MIX display or disassembled display mode), the memory window or the watch window while operating in the memory space expansion 4 MB mode, be aware that only bank 7 can be accessed. In this case, the data bank offset depends on the offset bits of the data bank register.
- When using the memory space expansion function 4 MB mode, execute the command for the memory space expansion function 4 MB mode to access each bank.

5. [Communication Baud Rate] Tab

Select communication baud rate between the E8a and MCU in the [Communication Baud Rate] tab. 2000000bps (default setting) should be selected.

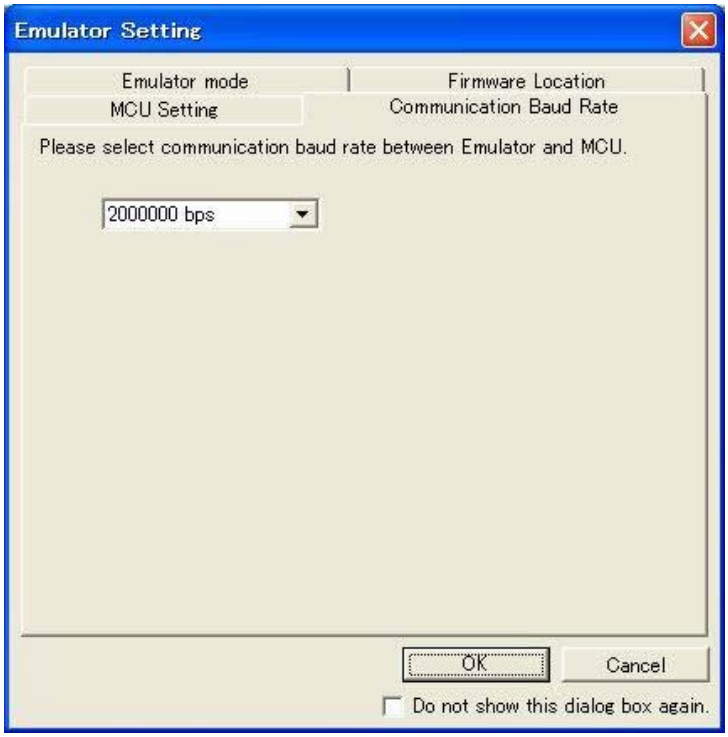


Figure 7.4 [Communication Baud Rate] Tab

Section 8 Command for Memory Space Expansion Function 4 MB Mode

1. Command for Memory Space Expansion Function 4 MB Mode

The following show the command for memory space expansion function 4 MB mode. These commands can be executed in the command line window.

Command	Description
Memory_Compare_Ext	Compares the memory area (between the start address and the end address) with the memory starting at destination address.
Memory_Display_Ext	Displays memory contents.
Memory_Fill_Ext	Fills an area of memory.
Memory_Find_Ext	Finds a string in a memory range.
Memory_Move_Ext	Moves memory.

2. Details of Command for Memory Space Expansion Function 4 MB mode

The following show the details of the command for memory space expansion function 4 MB mode.

Memory_Compare_Ext

Abbreviation: MCE

Description: Compares the memory area (between the start address and the end address) with the memory starting at destination address. This cannot be used during program execution.

Syntax: MCE <bank> <offsetbit> <start> <end> <destination> [<mode>]

Parameter	Type	Description
<bank>	Numeric	Bank (0 – 7)
<offsetbit> *	Numeric	Offset bit (0: no offset, 1: offset)
<start>	Numeric	Start address
<end>	Numeric	End address (including this address)
<destination>	Numeric	Destination address
<mode>	Keyword	Format (optional, default = BYTE)
	BYTE	1 byte
	WORD	2 bytes
	LONG	4 bytes
	DOUBLE	8 bytes

*: When “PM13 is 1” is not selected in the MCU setting dialog box, set the offset bit to “0”.

Memory_Display_Ext

Abbreviation: MDE

Description: Displays memory contents. This cannot be used during program execution.

Syntax: MDE <bank> <offsetbit> <address> [<length>] [<mode>]

Parameter	Type	Description
<bank>	Numeric	Bank (0 – 7)
<offsetbit> *	Numeric	Offset bit (0: no offset, 1: offset)
<address>	Numeric	Start address
<length>	Numeric	Length (optional, default = 0x100 bytes)
<mode>	Keyword	Display format (optional, default = BYTE)
	BYTE	Bytes
	WORD	Words (2 bytes)
	LONG	Long words (4 bytes)
	ASCII	ASCII
	SINGLE	Single-precision floating-point (4 bytes)
	DOUBLE	Double-precision floating-point (8 bytes)

*: When “PM13 is 1” is not selected in the MCU setting dialog box, set the offset bit to “0”.

Memory_Fill_Ext

Abbreviation: MFE

Description: Fills an area of memory. This cannot be used during program execution.

Syntax: MFE <bank> <offsetbit> <start> <end> <data> [<mode>] [<verify>]

Parameter	Type	Description
<bank>	Numeric	Bank (0 – 7)
<offsetbit> *	Numeric	Offset bit (0: no offset, 1: offset)
<start>	Numeric	Start address
<end>	Numeric	End address
<data>	Numeric	Data value
<mode>	Keyword	Data size (optional, default = BYTE)
	BYTE	Byte
	WORD	Word (2 bytes)
	LONG	Long word (4 bytes)
	ASCII	ASCII
	SINGLE	Single-precision floating-point (4 bytes)
	DOUBLE	Double-precision floating-point (8 bytes)
<verify>	Keyword	Verify flag (optional, default = V)
	V	Verification
	N	No verification

*: When “PM13 is 1” is not selected in the MCU setting dialog box, set the offset bit to “0”.

Memory_Find_Ext

Abbreviation: MIE
Description: Finds a string in a memory range. This cannot be used during program execution.
Syntax: MIE <bank> <offsetbit> <start> <end> <string> [<mode>]

Parameter	Type	Description
<bank>	Numeric	Bank (0 – 7)
<offsetbit> *	Numeric	Offset bit (0: no offset, 1: offset)
<start>	Numeric	Start address
<end>	Numeric	End address (including this address)
<string>	Numeric	String to search for
<mode>	Keyword	Format (optional, default = BYTE)
	BYTE	Bytes
	WORD	Words (2 bytes)
	LONG	Long words (4 bytes)
	ASCII	ASCII
	SINGLE	Single-precision floating-point (4 bytes)
	DOUBLE	Double-precision floating-point (8 bytes)

*: When “PM13 is 1” is not selected in the MCU setting dialog box, set the offset bit to “0”.

Memory_Move_Ext

Abbreviation: MVE
Description: Moves memory. This cannot be used during program execution.
Syntax: MVE <bank> <offsetbit> <start> <end> <destination> [<verify>] [<mode>]

Parameter	Type	Description
<bank>	Numeric	Bank (0 – 7)
<offsetbit> *	Numeric	Offset bit (0: no offset, 1: offset)
<start>	Numeric	Source start address
<end>	Numeric	Source end address (including this address)
<destination>	Numeric	Destination start address
<verify>	Keyword	Verify flag (optional, default = V)
	V	Verification
	N	No verification
<mode>	Keyword	Format (optional, default = BYTE)
	BYTE	1 byte
	WORD	2 bytes
	LONG	4 bytes
	DOUBLE	8 bytes

*: When “PM13 is 1” is not selected in the MCU setting dialog box, set the offset bit to “0”.

Section 9 Applicable Tool Chain and Third-party Products

With the M16C/62P and M16C/6N Group E8a emulator, you can debug modules created by the inhouse tool chain and third-party tools listed in Table 9.1 below.

Table 9.1 Applicable Tool Chain and Third-party Tools

Tool chain	M3T-NC30WA V.5.20 Release 01 or later
Third-party tools	TASKING M16C C/C++/EC++ Compiler V.2.3r1 or later IAR EWM16C V.2.12 or later

Notes on debugging the load modules created in ELF/DWARF2 format:
If the load module was created in ELF/DWARF2 format using TASKING M16C C/C++/EC++ compiler V3.0r1, the precautionary note described below must be observed when displaying member variables of the base class in the watch window.

Precautionary Note:
If any class object with a base class is defined, the following problems may occur:
Case 1: Member variables of the base class cannot be referenced directly from the class object (*1).
Case 2: If the PC value resides in any member function of a derived class, member variables of the base class cannot be referenced directly (*4).

Solution:
If member variables of the base class need to be referenced in the watch window, follow either method described below:
Case 1: Use indirect references from the class object to refer to member variables of the base class (*2) (*3).
Case 2: Use indirect references from “this” pointer to refer to member variables of the base class (*5) (*6).

<Example code>

```
//////////////////////////////////////////////////////////
*.h
    class BaseClass
    {
    public:
        int m_iBase;
    public:
        BaseClass() {
            m_iBase = 0;
        }
        void BaseFunc(void);
    };

    class DerivedClass : public BaseClass
    {
    public:
        int m_iDerive;
    public:
        DerivedClass() {
            m_iDerive   = 0;
        }
        void DerivedFunc(void);
    };
```

```
*.cpp
    main()
    {
        class DerivedClass ClassObj;
        ClassObj.DerivedFunc();
        return;
    }

    void BaseClass::BaseFunc(void)
    {
        m_iBase = 0x1234;
    }

    void DerivedClass::DerivedFunc(void)
    {
        BaseFunc();
        m_iDerive   = 0x1234;
    }
////////////////////////////////////////////////////////
```

< Watch window registration example>

```
////////////////////////////////////////////////////////
```

Case 1: If the PC value resides in the main() function

(1)"ClassObj.m_iBase" : Cannot be referenced (*1)

(2)"ClassObj.__b_BaseClass.m_iBase" : Can be referenced (*2)

(3)"ClassObj"

-"__b_BaseClass"

-"m_iBase" : Can be referenced (*3)

-"m_iDerive"

-: Expansion symbol

Case 2: If the PC value resides in the DerivedClass::DerivedFunc() function

(1)"m_iBase" : Cannot be referenced (*4)

(2)"this->__b_BaseClass.m_iBase" : Can be referenced (*5)

(3)"__b_BaseClass.m_iBase": Can be referenced (*5)

(4)"this"

-"*"

-"__b_BaseClass"

-"m_iBase" : Can be referenced (*6)

-"m_iDerive"

(5)"__b_BaseClass"

-"m_iBase" : Can be referenced (*6)

```
////////////////////////////////////////////////////////
```

E8a Emulator (R0E00008AKCE00)
Notes on Connecting the M16C/62P, M16C/6N4, M16C/6N5, M16C/6NK, M16C/6NM, M16C/6NL and M16C/6NN

Publication Date: Apr 30, 2010 Rev.4.00

Published by: Renesas Electronics Corporation

Edited by: Microcomputer Tool Development Department 2
Renesas Solutions Corporation

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: +44-1628-585-100, Fax: +44-1628-585-900

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 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

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.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

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.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2010 Renesas Electronics Corporation and Renesas Solutions Corporation. All rights reserved.

Colophon 1.0

E8a Emulator (R0E00008AKCE00)
Additional Document for User's Manual

Renesas Electronics Corporation

REJ10J1640-0400