

RTE-NB85E-CB

USER'S MANUAL (Rev. 1.05)

REVISION HISTORY

Date	Revision	Chapter	Explanation of revision
September 17, 1999	1.01		First edition
October 23, 1999	1.02	7.2.1	Revised BCC register value: 0x3C00 -> 0x3c40
January 17, 2000	1.03	7.2.1	Revised RFS3 register value: 0x0032/0016 -> 0x8032/8016
February 3, 2000	1.04	7.2.1	Revised RFS3 register value: 0x8032/8016 -> 0x8017/800F
October 6, 2000	1.05	5.4	Changed the function of a part of setup of SW-2 in accordance with corresponding to version .4 of CPU.

CONTENTS

1. INTRODUCTION	1
1.1. NUMERIC NOTATION	1
1.2. NOTES ON USE	1
2. FUNCTIONS	2
3. MAJOR FEATURES	3
4. BASIC SPECIFICATIONS	3
5. BOARD CONFIGURATION	4
5.1. RESET SWITCH (SW_RESET)	4
5.2. POWER CONNECTOR (JPOWER)	4
5.3. SWITCH 1 (SW1)	4
5.4. SWITCH 2 (SW2)	5
5.5. SWITCH 3 (SW3)	6
5.6. SWITCHES 11 TO 15 (SW11 TO 15)	7
5.7. SWITCH 16 (SW16)	8
5.8. SWITCH 17 (SW17)	8
5.9. 7SEG-LED, POWER-LED, and TOVER-LED	9
5.10. TEST PINS FOR ROM EMULATOR (JROM-EM1)	9
5.11. CLOCK SOCKET (OSC1)	9
5.12. CRYSTAL SOCKET (JP1)	9
5.13. VBCLK BUFFER SELECTOR JUMPER (JP2)	10
5.14. ROM SOCKETS	10
5.15. SERIAL CONNECTOR (JSIO1, JSIO2)	10
5.16. DEBUGGING CONNECTOR (JDCU)	11
5.17. JGBUS CONNECTOR (JGBUS)	11
5.18. CPU CONNECTOR (CN1 to CN4)	12
6. CONNECTION WITH THE HOST PC	16
6.1. RS-232C CONNECTION	16
7. HARDWARE REFERENCES	17
7.1. MEMORY AND I/O MAP	17
7.2. RECOMMENDED SETTINGS	19
7.2.1. MEMC Registers	19
7.3. MEMORY RESOURCES	20
7.3.1. SDRAM (CS1: 1000000 to 1FFFFFF)	20
7.3.2. SRAM (CS7: 3C00000 to 3FFFFFF)	20
7.3.3. UV-EPROM (CS0: 0000000 to 03FFFFFF)	20
7.3.4. Internal ROM (0000000 to 00FFFFFF, 0100000 to 01FFFFFF)	20
7.4. I/O MAP	21
7.5. I/O LIST	21
7.5.1. SW1 Read Port (SW1 3800000H [Read Only])	21
7.5.2. SW2 Read Port (SW2 3801000H [Read Only])	21
7.5.3. SW16 Read Port (SW16 3809000H [Read Only])	22

7.5.4.	7-Segment LED Display Data Output Port (7SEG-LED 3802000 [Write Only]).....	22
7.5.5.	Time-Over Ready LED Clear Pulse (TOVRDY_LED_CLRPLS 3803000H [Write Only]).....	22
7.5.6.	Interrupt Controller (PIC: 3804000 to 3804020 [Read/Write])	23
7.5.7.	UART (TL16C550C: 3807000 to 3807070)	24
7.5.8.	TIC (μ PD71054 3808000H to 380803FH).....	25
8.	SOFTWARE	26
8.1.	INITIALIZATION	26
8.2.	SUCCESSIVE ACCESSES TO μ PD71054.....	26
8.3.	LIBRARIES	26
8.4.	EXAMPLE OF USING TIMERS.....	27
9.	DEVELOPMENT OF APPLICATIONS USING MAS KABLE INTERRUPTS	28
9.1.	INTERRUPT VECTOR.....	28
9.2.	GENERAL RESTRICTIONS/NOTES.....	30
9.3.	REWRITING THE ALTERNATE VECTOR AREA DURING DOWNLOADING	30
9.4.	RESTRICTIONS/NOTES ON BREAKPOINTS	31
10.	CPU PIN CONNECTION.....	32
10.1.	LIST.....	32
10.2.	RESET-.....	33
10.3.	MWAIT-.....	34
10.4.	NMI0 TO NMI2 AND INTO	35
10.5.	INT10 TO INT13	36
10.6.	RXD/INT45	36
10.7.	TXD/INT46	36
10.8.	PORT0/MPXSCZ, PORT1/DSTBZ, PORT2/RDCYZ, PORT3/BUSST.....	37
10.9.	DMARQ0/INT32, DMARQ1/INT33, DMARQ2/INT34, DMARQ3/INT35	37
10.10.	DMAAK0/INT36, DMAAK1/INT37, DMAAK2/INT38, DMAAK3/INT39	38
10.11.	TC0/INT40, TC1/INT41, TC2/INT42, TC3/INT43	38
10.12.	OTHER SIGNALS	38
11.	SPECIFIC GBUS SPECIFICATIONS	39
11.1.	GENERAL.....	39
11.2.	BUS CYCLE	40
11.3.	CHIP SELECT.....	41
12.	APPENDIX A Multi MONITOR.....	42
12.1.	BOARD SETTING.....	42
12.1.1.	RTE for Win 32 Installation.....	42
12.1.2.	SW1 Setting	42
12.1.3.	Setting of Other Switches	42
12.1.4.	Connection of Board	42
12.2.	Multi MONITOR	43
12.2.1.	7-Segment LED on Startup.....	43
12.2.2.	ROM Monitor Work RAM.....	43
12.2.3.	Monitor Interrupt	43
12.2.4.	_INIT_SP Setting	43

12.2.5.	<i>Timer Interrupt</i>	43
12.2.6.	<i>Initializing Hardware</i>	43
12.2.7.	<i>Special Instruction</i>	44
12.3.	RTE COMMANDS	44
12.3.1.	<i>HELP(?)</i>	44
12.3.2.	<i>INIT</i>	44
12.3.3.	<i>VER</i>	44
12.3.4.	<i>SFR Command</i>	44
13.	APPENDIX B PARTNER MONITOR	45
13.1.	BOARD SETTING.....	45
13.1.1.	<i>SW1 Setting</i>	45
13.1.2.	<i>Setting of Other Switches</i>	45
13.1.3.	<i>Connection of Board</i>	45
13.2.	PARTNER MONITOR	46
13.2.1.	<i>7-Segment LED on Startup</i>	46
13.2.2.	<i>ROM Monitor Work RAM</i>	46
13.2.3.	<i>Monitor Interrupt</i>	46
13.2.4.	<i>SP Setting</i>	46
13.2.5.	<i>Initializing Hardware</i>	46
13.2.6.	<i>Special Instruction</i>	46
14.	APPENDIX C GBUS COMMON SPECIFICATIONS	47
14.1.	TERMINOLOGY	47
14.1.1.	<i>CPU Board and Motherboard</i>	47
14.1.2.	<i>Bus Cycle and Micro Cycle</i>	47
14.2.	SIGNALS	47
14.3.	PIN ASSIGNMENTS	52
14.4.	PROCESSING OF UNUSED PINS	53
14.5.	ALLOCATING GCS-[7:0]	53
14.6.	BUS CYCLE	54
14.6.1	<i>Single Cycle</i>	54
14.6.2.	<i>Burst Cycle</i>	54
14.6.3.	<i>GWAITI-</i>	55
14.6.4	<i>GBTERM-</i>	56
14.7.	TIMING	57
14.7.1.	<i>Setup Time</i>	57
14.7.2.	<i>Delay Time</i>	57

1. INTRODUCTION

The RTE-NB85E-CB is an evaluation board that is designed to evaluate the NEC NB85E (V850E core) RISC processor.

The board features an NB85E capable of operating at a maximum speed of 50 MHz, memory, serial interface, and bus connector for expansion. As the memories, a high-speed SRAM and high-capacity SDRAM are provided as standard. The SDRAM is controlled by using the internal memory controller of the NB85E.

These functions enable the RTE-NB85E-CB to be used for a wide variety of applications including processor performance evaluation and application program development at the initial stage, and to also be used as an engine for demonstration and simulation.

The GHS Multi or NEC PARTNER source-level debugger can be used as a development software tool with the RTE-NB85E-CB. The type of monitor to be stored in ROM depends on the debugger type.

In ROM, the monitor specified at the time of purchase is stored. Even when neither of the debuggers is purchased together with the RTE-NB85E-CB, they can be purchased at anytime subsequently.

1.1. NUMERIC NOTATION

This manual represents numbers according to the notation described in the following table. Hexadecimal and binary numbers may be hyphenated at every four digits, if they are difficult to read because of many digits being in each number.

Number	Notation rule	Example
Decimal number	Only numerals are indicated.	"10" represents number 10 in decimal.
Hexa-decimal number	A number is suffixed with letter H.	"10H" represents number 16 in decimal.
Binary number	A number is suffixed with letter B.	"10B" represents number 2 in decimal.

Number Notation Rules

1.2. NOTES ON USE

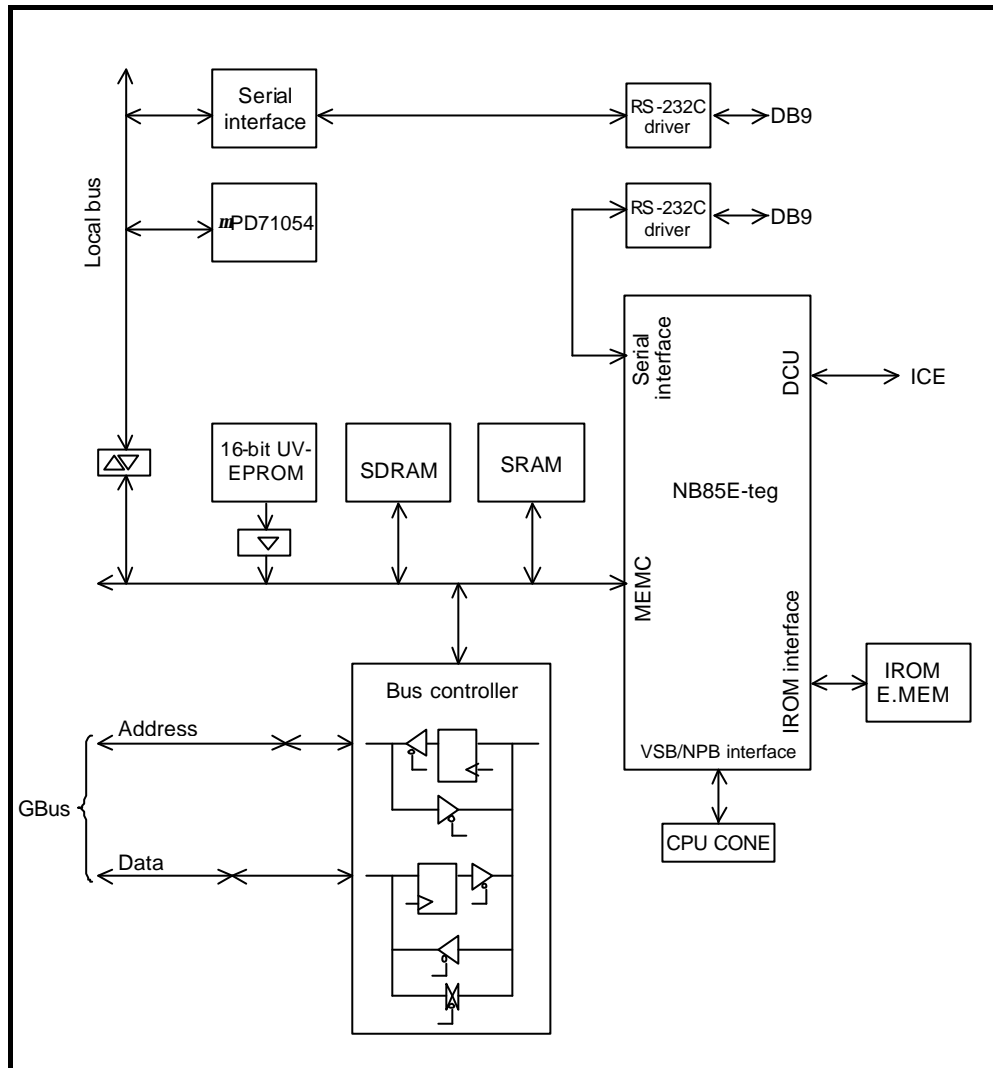
The NB85E chip to be mounted on this board is an evaluation sample and cannot be used for any purpose other than evaluation.

Note that the specifications of the board marked M883BA01 are different from those described in this manual.

Multi is a trademark of Green Hills Software, Inc. in the US.

2. FUNCTIONS

The overview of each function block of the RTE-NB85E-CB is shown below.



RTE-NB85E-CB Block Diagram

"Local bus" is a bus that buffers the MEMC bus and is synchronized with the CPU. "GBUS" is independent of the CPU and is fixed at 33 MHz.

3. MAJOR FEATURES

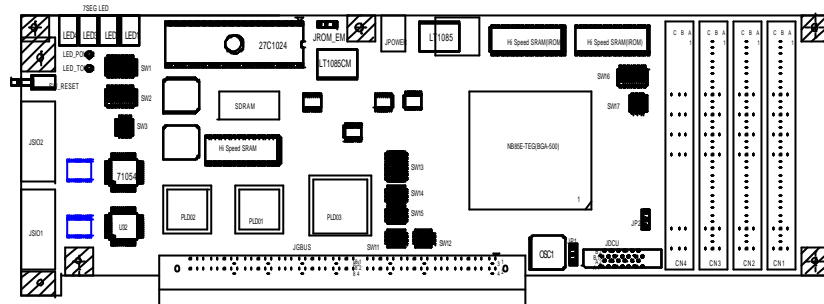
- Two types of monitor ROM are provided: one is used for the Green Hills Multi and the other for the NEC PARTNER.
- Real-time execution and evaluation at a high-level language level using Multi or PARTNER.
- Emulation memory of the internal ROM is provided.
- 1M byte of high-speed SRAM and 16M bytes of SDRAM (to access a 32-bit bus) are provided as standard.
- Two serial interfaces are provided. (One channel uses an external controller and the other channel uses the internal controller of the CPU. The serial channel of an external controller is used as the monitor.)
- Three timer channels are provided. (One channel is used for the monitor.)
- A ROM emulator can be connected.

4. BASIC SPECIFICATIONS

Processor	NB85E	
CPU clock	50 MHz	
Bus clock	50 MHz	
Power consumption	+5 V (2 A)	
Memory		
EPROM	128 KB	64 K × 16 bits (40-pin DIP) × 1 (512K bytes max.)
IROM-RAM	1 MB	256 K × 8 bits × 4
SRAM	1 MB	256 K × 16 bits × 2
SDRAM	16 MB	1 M × 16 bits × 4 banks × 2
I/O		
Serial (2 ch)	Internal CPU, DB9 connector Equivalent to NS16550, DB9 connector	
Timer	Equivalent to i8254, 500-ns resolution	
I/O port	LED (7-segment) × 4, display/switch input	
Others		
CPU connector	Connector with all function pins of the NB85E connected (except the pins for IROM interface)	
32-bit standard external extension bus	RTE-CB standard 32-bit interface (4G bytes, 32-bit bus, correspond to DMA)	
Reset switch	Push type	

5. BOARD CONFIGURATION

The physical layout of the major components on the RTE-NB85E-CB board is shown below. This chapter explains each component.



RTE-NB85E-CB Components Layout

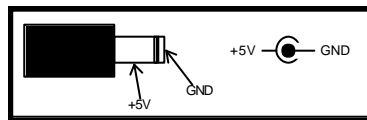
5.1. RESET SWITCH (SW_RESET)

SW_RESET is a reset switch for the entire board. Pressing this switch causes all the circuits including the CPU to be reset.

5.2. POWER CONNECTOR (JPOWER)

The power supplied to the JPOWER connector should be one rated as listed below.

- Voltage: +5 V
- Current: Maximum of 2 A
- Mating connector: Type A (5.5 mm in diameter)
- Polarity:



Use only the special power supply (RTE-PS01) supplied. To supply power from the JBUS connector, do not connect a power source to JPOWER.

5.3. SWITCH 1 (SW1)

SW1 is a general-purpose input port switch. The setting status can be read from an input port (see Section 7.5.1). When the port is read, a switch being set to OFF represents 1, while its being set to ON represents 0. When the monitor ROM is used, all SW1 switches except some are already set. Set this switch for assignment with the monitor ROM by referring to the following sections and in accordance with your environment:

- When using Multi, see Section 12.1.2.
- When using PARTNER, see Section 13.1.1.

5.4. SWITCH 2 (SW2)

SW2 selects an operation of the board. The setting of the switch can be read from an input port (see Section 7.5.2).

No.	Signal name	Factory setting	Function
1	FBOOT	OFF	Specifies resources to be allocated to the CS0 space. OFF: The on-board UV -EPROM is allocated to the CS0 space. ON: GCS1- space of GBUS is allocated to the CS0 space (see Section 7.1).
2	TEST	OFF	Set this signal to OFF.
3	BCLK_LOW	OFF	Selects frequency of oscillator mounted on OSC1. Depending on the value set, the monitor ROM changes the number of ROM and SRAM wait cycles. In addition, the number of I/O wait cycles is changed by hardware. OFF: Bus clock exceeds 33 MHz. ON: Bus clock is kept at 33 MHz or less.
4	BSIZE16	OFF	Selects bus width on SRAM and SDRAM hardware. OFF: 32 bits wide ON: 16 bits wide (MEMC must be set in accordance with this setting.)
5	NM/INT0-	ON	Specifies interrupt to be used by the monitor. OFF: NMI ON: INT0
6	CACHE	OFF	Specifies initialization of the cache. OFF: Initializes all spaces with UNCACHE. ON: Initializes all on-board memory resources with CACHE.
7			It changes with versions of monitor ROM.
8			The following should do explanation reference.

When the version of monitor ROM is 2.xx(for CPU of Ver.3), please set up according to the following.

7	SCPUMODE0	OFF	Performs the following setting in accordance with mode specified by SW16: SW16 [SCPUMODE1, SCPUMODE0]
8	SCPUMODE1	OFF	AUTO [OFF , OFF] <<Normal setting SINGLE MODE0 [OFF , ON] SINGLE MODE1 [ON , OFF] ROMLESS [ON , ON]

[Caution] SW2-7 and SW2-8 select a monitor mode that matches the CPU mode. Usually, select AUTO.

When the version of monitor ROM is 3.xx(for CPU of Ver.4), please set up according to the following.

7	CACHEMODE 0	ON	The mode in case SW 2-6 is ON (CACHE ON) is specified.SW16: OFF: Cached Write -through ON: Cached Write-back
8	CACHEMODE 1	OFF	Wright allocation in case SW 2-7 is ON (Write-back mode) is specified. OFF: Disable ON: Enable

5.5. SWITCH 3 (SW3)

SW3 selects the type of ROM inserted in the ROM socket and performs setting related to banks.

No.	Signal name	Factory setting	Function
1	ROM_TYPE0	OFF	Selects the type of ROM. [ROM_TYPE1, ROM_TYPE0]
2	ROM_TYPE1	OFF	[OFF , OFF]: When monitor ROM is used [OFF , ON]: When 27C4096 is used [ON , OFF]: When 27C2048 is used [ON , ON]: When 27C1024 is used
3	BANK_DIS	OFF	Specifies whether the upper and lower halves (banks) of ROM are separated. Be sure to set this signal to OFF when monitor ROM is used. OFF: Upper and lower halves of ROM are separated. ON: Upper and lower halves of ROM are used as a contiguous area.
4	BANK_LOW	OFF	Selects either the upper or lower half when SW3-3 is OFF. OFF: Selects upper half. ON: Selects lower half.

[Caution] The monitor ROM has a code by which it operates in ROMLESS mode (and SINGLE MODE1 mode) in the lower half, and a code by which it operates in SINGLE MODE0 mode in the upper half. To change the setting of SW16 (mode setting of CPU), change SW3-4 to the appropriate setting.

5.6. SWITCHES 11 TO 15 (SW11 TO 15)

SW11-15 physically cuts the board's signal lines connected to CPU pins. All the switches are factory-set to ON (connected). Set a switch to OFF only when the corresponding signal line is used for an external source, but only if the internally used resources are not necessary.

Remark The following tables show the CPU pins and final internal resource names.

[SW11]

No.	CPU pin name	Factory setting	Internally used resource
1	NMI0	ON	Interrupt from ROM emulator
2	NMI1	ON	Interrupt request signal GINT0- of GBUS
3	NMI2	ON	Monitor interrupt (NMI)
4	INT0	ON	Monitor interrupt (INT0)

[SW12]

No.	CPU pin name	Factory setting	Internally used resource
1	PORT0/MPXCSZ	ON	CTS- of JSIO2
2	PORT1/DSTBZ	ON	DSR- of JSIO2
3	PORT2/RDCYZ	ON	RTS- of JSIO2
4	PORT3/BUSST	ON	DTR- of JSIO2

[SW13]

No.	CPU pin name	Factory setting	Internally used resource
1	INT32/DMARQ0/TBO0	ON	DMARQ0- of GBUS
2	INT33/DMARQ1/TBO1	ON	DMARQ1- of GBUS
3	INT34/DMARQ2/TBO2	ON	DMARQ2- of GBUS
4	INT35/DMARQ3/TBO3	ON	DMARQ3- of GBUS
5	INT36/DMAAK0/TBO4	ON	DMAAK0- of GBUS
6	INT37/DMAAK1/TBO5	ON	DMAAK1- of GBUS
7	INT38/DMAAK2/TBO6	ON	DMAAK2- of GBUS
8	INT39/DMAAK3/TBO7	ON	DMAAK3- of GBUS

[SW14]

No.	CPU pin name	Factory setting	Internally used resource
1	INT40/TC0/TBO8	ON	Pin 129 of GBUS
2	INT41/TC1/TBO9	ON	Pin 130 of GBUS
3	INT42/TC2/TBO10	ON	Pin 131 of GBUS
4	INT43/TC3/TBO11	ON	Pin 132 of GBUS
5	INT45/RXD/TBO13	ON	RXD (receive data) of JSIO2
6		OFF	No use
7		OFF	NO use
4		OFF	No use

[SW15]

No.	CPU pin name	Factory setting	Internally used resource
1	INT10/TBI10	ON	GINT1- of GBUS
2	INT11/TBI11	ON	GINT2- of GBUS
3	INT12/TBI12	ON	GINT3- of GBUS
4	INT13/TBI13	ON	OUT1 of TIC

5.7. SWITCH 16 (SW16)

SW16 specifies the status of the MODE[0..6] pins of the CPU. When a signal of this switch is OFF, the corresponding MODE pin of the CPU is 1; when it is ON, the pin is 0.

The set value is shown below.

Initial Bus Wide = 32-bit

No.	CPU pin name	Single Mode0	Single Mode1	Romless
1	MODE0	ON	OFF	ON
2	MODE1	ON	ON	ON
3	MODE2	ON	ON	ON
4	MODE3	OFF	OFF	ON
5	MODE4	OFF	OFF	OFF
6	MODE5	OFF	OFF	OFF
7	MODE6	OFF	OFF	OFF
8	ROM16 (Note 2)	OFF	OFF	OFF

Initial Bus Wide = 16-bit

No.	CPU pin name	Single Mode0	Single Mode1	Romless
1	MODE0	ON	OFF	ON
2	MODE1	ON	ON	OFF
3	MODE2	OFF	OFF	OFF
4	MODE3	OFF	OFF	ON
5	MODE4	OFF	OFF	OFF
6	MODE5	OFF	OFF	OFF
7	MODE6	OFF	OFF	OFF
8	ROM16 (Note 2)	OFF	OFF	OFF

<<Notes>>

1. The factory setting for shipment is Initial Bus Wide = 32-bit, Single Mode0. Usually, use this setting.
2. SW16-8:ROM16 is not used.
3. When the CPU mode is changed, the type of monitor must be changed accordingly.
Set SW3-4:BANK_LOW to OFF in Single Mode0, and to ON in any other mode (see Section 5.5).

5.8. SWITCH 17 (SW17)

SW17 specifies the status of a CPU pin. When a signal of this switch is set to OFF, the corresponding CPU pin is 1; when it is set to ON, the pin is 0.

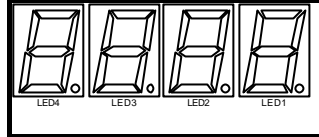
No.	CPU pin name	Factory setting	Function
1	CKSEL0	ON	CPU clock frequency is two times the frequency of OSC1.
2	CKSEL1	ON	
3	VOESEL	OFF	VSB Output Signal becomes valid immediately after reset.
4	ROMAA2	OFF	Must be OFF

<<Note>>

Usually, do not change the factory setting of this switch.

5.9. 7SEG-LED, POWER-LED, AND TOVER-LED

The LEDs are used to indicate statuses, as listed below. The four 7-segment LEDs are used by the monitor at startup. After that, they can be used for any user application.



LED	Description
POWER	Lights when power is supplied to the RTE-NB85E-CB board.
TOVRDY	Lights when time-over ready occurs, and does not go off until cleared by software (see Section 7.5.5).

Board LED Status

5.10. TEST PINS FOR ROM EMULATOR (JROM-EM1)

Test pins (JROM-EM1s) are used to connect a ROM emulator. They accept control signals listed below. The following table lists the signal names and functions.

Signal	Input/output	Function
RESET- (1)	Input	When a low level is supplied to this test pin, the CPU is reset. A reset request signal from the ROM emulator is connected to the test pin. The test pin is pulled up with 1 k Ω .
NMI- (2)	Input	When a low level is supplied to this test pin, an NMI signal is given to the CPU. (See Section 10.4.) An NMI request signal from the ROM emulator is connected to the test pin. The test pin is pulled up with 1 k Ω .
GND (3)	---	This test pin is at a ground level. The ground level of the ROM emulator is connected to the test pin.

JROM-EM1 Pin Functions

5.11. CLOCK SOCKET (OSC1)

An oscillator for generating the clock signal to be supplied to the CPU is mounted in the OSC1 socket. OSC1 is converted to the 3.3-V level, and is connected to the CPUCLK pin of the CPU.

Accepts DIP 8-pin (half-type) oscillators.



When you have to cut an oscillator pin for convenience, be careful not to cut it too short, or otherwise the frame (housing) of the oscillator may touch a pin in the socket, resulting in a short-circuit occurring.

5.12. CRYSTAL SOCKET (JP1)

JP1 has two functions: it is used to select the clock supplied to the CPU and it also acts as the connector for the crystal oscillator.

To use OSC1 as CPU clock

Short-circuit JP1 pins 1 and 2. In this case, do not mount the crystal.

To mount crystal on JP1 and use the CPU oscillation circuit

Mount the crystal between pins 1 and 3 on JP1. Do not short-circuit pins 1 and 2. If oscillation is unstable, adjust C10 and C11 (10 pF is connected as a factory-set condition).

5.13. VBCLK BUFFER SELECTOR JUMPER (JP2)

JP2 is a jumper that selects the type of the buffer for clock output to CNx.

If pins 1 and 2 are short-circuited: The signal buffered by 74LVTH541 is output (delay time: several ns).

If pins 2 and 3 are short-circuited: The signal buffered by CY2308 is output (delay time: zero).

Remark If the delay time poses a problem, use CY2308. Because PLL is used, however, it is not suitable if the clock is varied or stopped.

5.14. ROM SOCKETS

The RTE-NB85E-CB has ROM sockets to hold 40-pin ROM chips to provide standard 128K bytes (64K×16 bits). When the ROM chips used here are to be replaced, their type should be 27C1024, 27C2048, or 27C4096, and the access time should be 120 ns or less.

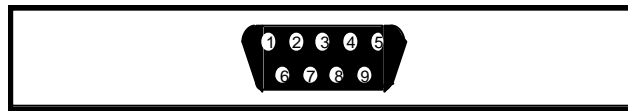
5.15. SERIAL CONNECTOR (JSIO1, JSIO2)

The JSIO1 connector is used for the RS-232C interface that is controlled by the serial controller (TL16C550C) on the board.

The JSIO2 connector is used for the RS-232C interface that is controlled by the built-in serial controller of the CPU.

JSIO1 and JSIO2 are 9-pin D-SUB RS-232C connectors (male) like that commonly used on the PC/AT. All signals on both of these connectors are converted to the RS-232C level. The figure and table below indicate the pin and signal arrangement of these connectors.

For the signals to be connected to the host, the table indicates two modes of wiring on the host: one for a 9-pin D-SUB connector, and the other for a 25-pin D-SUB connector. (Regular cross-cable wiring is used for these connections.)



Pin Arrangement of JSIO1 and JSIO2 (Male)

JSIO1 pin No.	JSIO2 pin No.	Signal name	Input/ output	Connector pin number on the host side	
				D-SUB9	D-SUB25
1	¹ Note 1	DCD	Input		
2	2	RxD(RD)	Input	3	2
3	3	TxD(SD)	Output	2	3
4	4	DTR(DR)	Output	1, 6	6, 8
5	5	GND		5	7
6	6	DSR(ER)	Input	4	20
7	7	RTS(RS)	Output	8	5
8	8	CTS(CS)	Input	7	4
9	⁹ Note 1	RI	Input		

JSIO1 and JSIO2 Connector Signals

Notes:

1. JSIO2 pins 1 and 9 are not used on board.
2. JSIO2 uses UART of NB85E-TEG and PORT3..0. For the connection status, see Sections 10.6 to 10.8.

5.16. DEBUGGING CONNECTOR (JDCU)

The JDCU connector is used to connect a debug tool based on the debug function built into the NB85E.

Pin No.	Signal name	Pin No.	Signal name
A1	TRCLK	B1	GND
A2	TRCDATA0	B2	GND
A3	TRCDATA1	B3	GND
A4	TRCDATA2	B4	GND
A5	TRCDATA3	B5	GND
A6	TRCEND	B6	GND
A7	DDI	B7	GND
A8	DCK	B8	GND
A9	DMS	B9	GND
A10	DDO	B10	GND
A11	DRST-	B11	NC.
A12	NC.	B12	NC.
A13	NC.	B13	+3.3V

JDCU Connector Signals

On-board connector: 8830E-026-170S manufactured by KEL

5.17. JGBUS CONNECTOR (JGBUS)

This is a 32-bit bus connector for expansion. For details, see Chapters 11 and 14.

5.18. CPU CONNECTOR (CN1 TO CN4)

The CPU connector signals are connected directly to the NB85E. Many signals of the MEMC bus are used on the board. So, be careful when extracting signals from the JCPU. The 3.3-V signal level is used. Because a DIN96-pin connector can be connected, connect a connector suitable for your application (no connector is connected as a factory-set condition).

	Signal name: column A	Signal name: column B	Signal name: column C
1	+5V	GND	VBD0
2	VBD1	VBD2	VBD3
3	VBD4	VBD5	VBD6
4	VBD7	GND	VBD8
5	VBD9	VBD10	VBD11
6	VBD12	VBD13	VBD14
7	VBD15	GND	VBD16
8	VBD17	VBD18	VBD19
9	VBD20	VBD21	VBD22
10	VBD23	GND	VBD24
11	VBD25	VBD26	VBD27
12	VBD28	VBD29	VBD30
13	VBD31	+5V	GND
14	VBA0	VBA1	VBA2
15	VBA3	VBA4	VBA5
16	VBA6	VBA7	GND
17	VBA8	VBA9	VBA10
18	VBA11	VBA12	VBA13
19	VBA14	VBA15	GND
20	VBA16	VBA17	VBA18
21	VBA19	VBA20	VBA21
22	VBA22	VBA23	VBA24
23	VBA25	VBA26	VBA27
24	GND	(NC)	(NC)
25	VAREQ0	VBWAIT	VBAHLD
26	VBLAST	VBEXDC	VBEXCLK
27	GND	RESET-	VAACK0
28	VBLOCK	VBSTZ	VBWRITE
29	GND	VBCTYP0	VBCTYP1
30	VBCTYP2	VBTTYP0	VBTTYP1
31	GND	VBSTR	VBSEQ0
32	VBSEQ1	VBSEQ2	+3.3V

CN1 Connector Signals

	Signal name: column A	Signal name: column B	Signal name: column C
1	+5V	GND	VBBENZ0
2	VBBENZ1	VBBENZ2	VBBENZ3
3	VBSIZE0	VBSIZE1	VBDC
4	VDSELPZ	GND	VDCSZ0
5	VDCSZ1	VDCSZ2	VDCSZ3
6	VDCSZ4	VDCSZ5	VDCSZ6
7	VDCSZ7	GND	VBCLK1
8	GND	NMI0	NMI1
9	NM2	GND	INT0/TBI0
10	INT1/TBI1	INT2/TBI2	INT3/TBI3
11	INT4/TBI4	INT5/TBI5	INT6/TBI6
12	INT7/TBI7	INT8/TBI8	INT9/TBI9
13	INT10/TBI10	+5V	GND
14	INT11/TBI11	INT12/TBI12	INT13/TBI13
15	INT14/TBI14	INT15/TBI15	INT16/TBI16
16	INT17/TBI17	INT18/TBI18	INT19/TBI19
17	INT20/TBI20	INT21/TBI21	INT22/TBI22
18	INT23/TBI23	INT24/TBI24	INT25/TBI25
19	INT26/TBI26	INT27/TBI27	INT28/TBI28
20	INT29/TBI29	INT30/TBI30	INT31/TBI31
21	GND	INT32/DMARQ0/TBO0	INT33/DMARQ1/TBO1
22	INT34/DMARQ2/TBO2	INT35/DMARQ3/TBO3	INT36/DMAAK0/TBO4
23	INT37/DMAAK1/TBO5	INT38/DMAAK2/TBO6	INT39/DMAAK3/TBO7
24	INT40/TC0/TBO8	INT41/TC1/TBO9	INT42/TC2/TBO10
25	INT43/TC3/TBO11	INT44/DMASTP/TBO12	INT45/RXD/TBO13
26	INT46/TXD/TBO14	GND	INT47/TBO15/TAPSM0
27	INT48/TBO16/TAPSM1	INT49/TBO17/TAPSM2	INT50/TBO18/TAPSM3
28	INT51/ELKRT/TBO19/ DDOJT	INT52/EINTLV0/TBO20/ DDOENB	INT53/EINTLV1/TBO21/ DBRESZ
29	INT54/EINTLV2/TBO22/ RESMK	INT55/EINTLV3/TBO23/ MSKSTP	INT56/RINTLV4/TBO24/ MSKNMI0
30	INT57/EINTLV5/TBO25/ MSKNMI1	INT58/EINTLV6/TBO26/ MSKNMI2	INT59/EINTRQ/TBO27/ MSKHRQ
31	INT60/EINTAK/TBO28/ DBRDY	INT61/EASTB/TBO29/ EVASTB	INT62/EDSTB/TBO30/ EVDSTB
32	INT63/ECLRIP/TBO31/ EVCLRIP	GND	+3.3V

CN2 Connector Signals

	Signal name: column A	Signal name: column B	Signal name: column C
1	+5V	GND	VPD0/EAD0
2	VPD1/EAD1	VPD2/EAD2	VPD3/EAD3
3	VPD4/EAD4	VPD5/EAD5	VPD6/EAD6
4	VPD7/EAD7	GND	VPD8/EAD8
5	VPD9/EAD9	VPD10/EAD10	VPD11/EAD11
6	VPD12/EAD12	VPD13/EAD13	VPD14/EAD14
7	VPD15/EAD15	GND	VPA0
8	VPA1	VPA2	VPA3
9	VPA4	VPA5	VPA6
10	VPA7	VPA8	VPA9
11	VPA10	VPA11	VPA12
12	VPA13	(NC)	(NC)
13	(NC)	+5V	GND
14	VPRETR	VPDACT	VPSTB
15	VPWRITE	VPLOCK	VPUBENZ
16	VPEXRETR	VPEXDACT	GND
17	TBI32	TBI33	TBI34
18	TBI35	TBI36	TBI37
19	TBI38	TBI39	TEST
20	BUNRI	GND	TBO33
21	TBO32	TBO34	VPTCLK
22	PHTEST	TESTN	TBREDZ
23	GND	DBINT	EVTTRG
24	IDBR0	IDBR1	IDBR2
25	(NC)	(NC)	(NC)
26	(NC)	(NC)	(NC)
27	RESET_VB-	(NC)	IORD-
28	IOWR-	HLDAK-	HLDREQ-
29	GND	(NC)	(NC)
30	(NC)	STOPZ-	PWM/ASTBZ
31	PORT0/MPXC SZ	PORT1/DSTBZ	PORT2/RDCYZ
32	PORT3/BUSST	GND	+3.3V

CN3 Connector Signals

	Signal name: column A	Signal name: column B	Signal name: column C
1	+5V	GND	D0
2	D1	D2	D3
3	D4	D5	D6
4	D7	GND	D8
5	D9	D10	D11
6	D12	D13	D14
7	D15	GND	D16
8	D17	D18	D19
9	D20	D21	D22
10	D23	GND	D24
11	D25	D26	D27
12	D28	D29	D30
13	D31	+5V	GND
14	A0/EVAD0	A1/EVAD1	A2/EVAD2
15	A3/EVAD3	A4/EVAD4	A5/EVAD5
16	A6/EVAD6	A7/EVAD7	A8/EVAD8
17	A9/EVAD9	A10/EVAD10	A11/EVAD11
18	A12/EVAD12	A13/EVAD13	A14/EVAD14
19	A15/EVAD15	GND	A16/EVLKRT
20	A17/EVINTLV0	A18/EVINTLV1	A19/EVINTLV2
21	A20/EVINTLV3	A21/EVINTLV4	A22/EVINTLV5
22	A23/EVINTLV6	A24/EVINTRQ	A25/EVINTAK
23	GND	CS0-/RAS0-	CS1-/RAS1-
24	CS2-/RAS2-	CS3-/RAS3-	CS4-/RAS4-
25	CS5-/RAS5-	CS6-/RAS6-	CS7-/RAS7-
26	GND	VBCLK1 ^{Note}	GND
27	BCYST-	RD-	WR0-/CAS0-/DQM0-
28	WR1-/CAS1-/DQM1-	WR2-/CAS2-/DQM2-	WR3-/CAS3-/DQM3-
29	GND	OE-	WE-
30	REFRQ-	SDRAS-	SDCAS-
31	CKE	GND	SDCLK
32	GND	MWAIT-	+3.3V

CN4 Connector Signals

Note VBCLK1 is a signal generated by buffering VBCLK.

6. CONNECTION WITH THE HOST PC

6.1. RS-232C CONNECTION

Serially connect the host machine using the monitor ROM by means of the following procedure:

- <1> Get an optional RS-232C cable and a power supply.
- <2> Set and check the setting of the switches on the board. Specify a baud rate by using SW1 (see Sections 12.1.2 and 13.1.1).
- <3> Connect the JSIO1 connector and host machine with the RS-232C cable, and supply power to the JPOWER connector. Confirm that the POWER-LED on the board lights and that the 7-segment LED indicating that the monitor has started lights.



If the LED does not light, turn off the power immediately, and check the connection.

- <4> Start the debugger on the host machine, and connect it via the RS-232C interface. If an error occurs, confirm the type of the monitor and the setting of the switches (baud rate and CPU mode). For the method and procedure of starting the debugger, see the debugger manual.



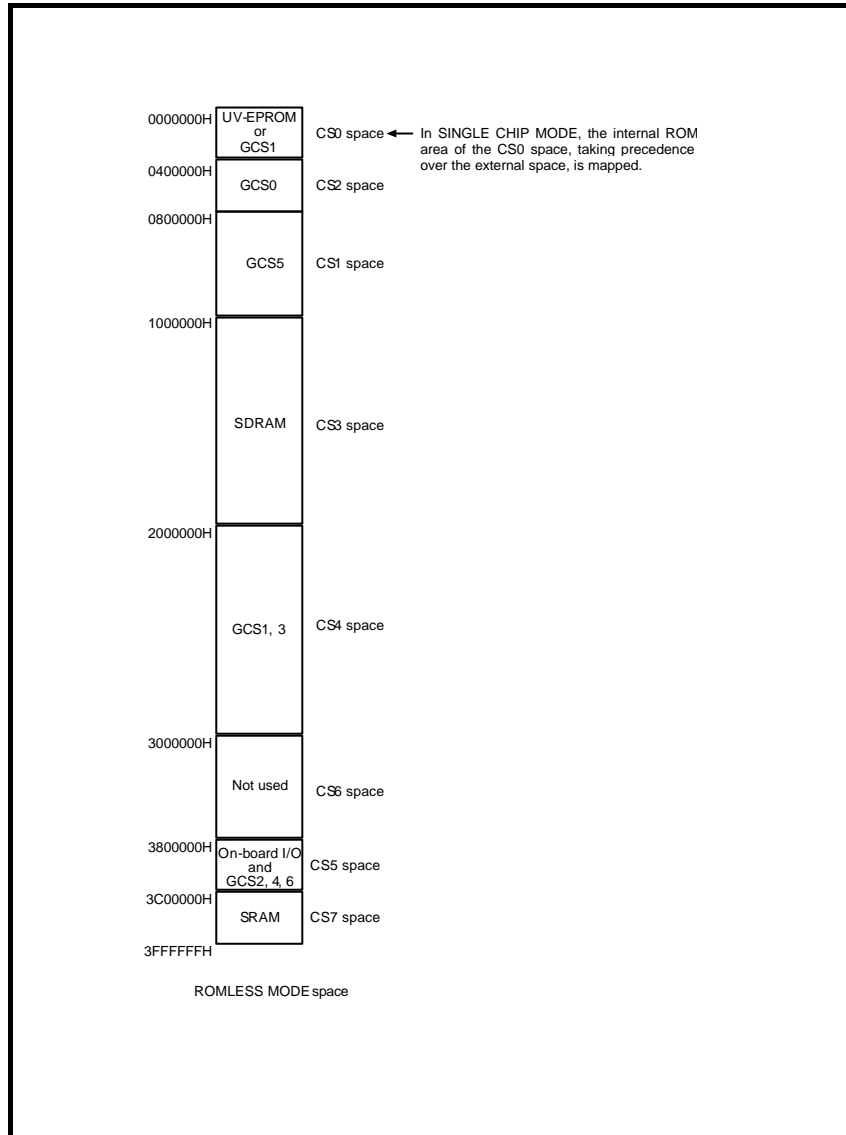
Place the board on an insulating material. If a conductive material touches the board while power is supplied to the board, the board may malfunction.

7. HARDWARE REFERENCES

This chapter describes the hardware of the RTE-NB85E-CB.

7.1. MEMORY AND I/O MAP

The figure below shows the memory and I/O mapping on the board. The address mode is 64M mode.



Memory I/O Map

CS0 space (UV-EPROM, GCS1-): 0000000 to 03FFFFFF (4M bytes)

UV-EPROM is allocated to the CS0 space, except the internal ROM space in SingleChip mode, or the CS0 space is reserved as a space for GCS-1 of GBUS. On-board UV-EPROM is allocated to the space when SW2-2 (FBOOT) is OFF (see Section 5.4). When SW2-2 (FBOOT) is ON, it is reserved as a space for GCS1- of GBUS. The GCS1 space can be accessed from the CS4 space. By locating a flash ROM in the GCS1 space of GBUS, therefore, the flash ROM can be rewritten by using the monitor ROM of UV-EPROM. Thereafter, a program can be booted from the flash ROM.

For GBUS, see Chapters 11 and 14.

If SingleChip mode is set, an emulation memory takes precedence over other memories in allocation to the internal ROM space (which starts from 0x0- or 0x100000-).

CS1 space (GCS5-): 0800000 to 0FFFFFFF (8M bytes)

The CS1 space is reserved as a space for GCS5- of GBUS. For GBUS, see Chapters 11 and 14.

CS2 space (GCS0-): 0400000 to 07FFFFFF (4M bytes)

The CS2 space is reserved as a space for GCS0- of GBUS. For GBUS, see Chapters 11 and 14.

CS3 space (SDRAM): 1000000 to 1FFFFFFF (16M bytes)

SDRAM is allocated to the CS3 space. The bus size of this space can be changed between 16 bits and 32 bits, depending on the setting of SW2-4 (BSIZE). If monitor ROM is mounted, the monitor ROM allocates and initializes SDRAM.

CS4 space (GCS1-, GCS3-): 2000000 to 2FFFFFFF (16M bytes)

The CS4 space is reserved as an area for GCS1- and GCS3- of GBUS. For GBUS, see Chapters 11 and 14.

CS5 space (I/O, GCS2-, GCS4-, GCS6-): 3800000 to 3BFFFFFF (4M bytes)

The CS5 space is used as an I/O space. There are reserved spaces for the board's I/O and GCS2-, GCS4-, and GCS6- of GBUS. For the I/O map, see Section 7.4. For GBUS, see Chapters 11 and 14.

CS6 space (not used): 3000000 to 37FFFFFF (8M bytes)

The CS6 space is not used on the board.

CS7 space (SRAM) 3C00000 to 3FFFFFFF (4M bytes)

SRAM is allocated to the CS7 space. The bus size of this space can be changed between 16 bits and 32 bits, depending on the setting of SW2-4 (BSIZE). If monitor ROM is mounted, the monitor ROM initializes SRAM. Because the 32-KB second-half area of the SRAM is used by the monitor as a work area, it cannot be used by the user program (see Sections 12.2.3 and 13.2.2).

7.2. RECOMMENDED SETTINGS

This section explains the recommended setting values of each register related to memory and I/O resource access.

7.2.1. MEMC Registers

Make the following register settings for the system bus. Note that the setting for some registers differs depending on the settings of SW2-4 (BSIZE) and SW2-3 (BCLK_LOW).

Register name	Address	Setting	Remarks
BCT0	0xFFFF480	0xB888	CS0 to CS2: SRAM/IO, CS3: SDARM
BCT1	0xFFFF482	0x8888	CS4 to CS7: SRAM/IO
DWC0 (BCLK_LOW = OFF) DWC0 (BCLK_LOW = ON)	0xFFFF484	0x1111 0x0000	CS0 to CS3: 1 wait CS0 to CS3: 0 wait
DWC1 (BCLK_LOW = OFF) DWC1 (BCLK_LOW = ON)	0xFFFF486	0x1711 0x0700	CS4 to CS5, CS7: 1 wait, CS6: 7 waits CS4 to CS5, CS7: 0 wait, CS6: 7 waits
VSWC (BCLK_LOW = OFF) VSWC (BCLK_LOW = ON)	0xFFFF06E	0x77	At present, the default value (0x77) regardless of status of BCLK_LOW
BCC	0xFFFF488	0x3C40	CS0,1,2,4, 7: 0 clk, CS3:1clk, CS5, CS6:3 clks
ASC	0xFFFF48A	0x5555	All 1 addr wait
BCP	0xFFFF48C	0x00	Normal bus cycle
CSC0	0xFFFF060	0xCCC3	(Chip Select Control Register0)
CSC1	0xFFFF062	0xCCC3	(Chip Select Control Register1)
BSC (BSIZE= OFF) BSC (BSIZE= ON)	0xFFFF066	0xAAAA 0x6A6A	All 32-bit Only CS3, CS7 16 bits
BEC	0xFFFF068	0x0000	All little endian
SCR3 (BSIZE= OFF) SCR3 (BSIZE= ON)	0xFFFF4AC	0x20A4 0x2094	SDRAM (LTM = 2, BCW = 2, SSO = 2, RAW = 12, SAW = 0) SDRAM (LTM = 2, BCW = 2, SSO = 1, RAW = 12, SAW = 0)
RFS3 (BCLK_LOW = OFF) RFS3 (BCLK_LOW = ON)	0xFFFF4AE	0x8017 0x800F	50 MHz: 15.4 ns 33 MHz: 15.5 ns
ISS	0xFFFF7FA	0x4F	ISS0 = 1, ISS1 = 1, ISS = 2 = 1, ISS3 = 1, ISS6 = 1
RSZ	0xFFFF7FC	0x40	IRAM = 60 KB
DCC	0xFFFF078	0x0C00	DC05:04 = WriteBack (WriteAllocate Enable)
CSZ	0xFFFF7FE	0x10	DTYP = 1 (SDRAM)
BHC (CACHE = OFF) BHC (CACHE = ON)	0xFFFF06A	0x0000 0xC0C0	All uncached CS7, CS3 i/d cacheable

* For CACHE = OFF/ON, use SW2-6.

[Caution] For the setting procedure of SDRAM-related registers (SCR3 and SFR3), see the CPU manual.

7.3. MEMORY RESOURCES

RTE-NB85E-CB has SDRAM, SRAM, and UV-EPROM as on-board memory resources. This section explains these memory devices.

7.3.1. SDRAM (CS3: 1000000 to 1FFFFFFF)

Two SDRAM devices (μ PD4564163G5), each consisting of 1M word \times 16 bits \times 4 banks, are provided as SDRAM. The total capacity is 16M bytes. The data bus width can be set to 16 or 32 bits, depending on the setting of SW2-4 (BSIZE). If a 16-bit bus width is used, however, the memory capacity is halved to 8M bytes.

7.3.2. SRAM (CS7: 3C00000 to 3FFFFFFF)

Two high-speed SRAM devices, each having a capacity of 256K words \times 16 bits and a speed of 15 ns, are provided as SRAM. The total SRAM capacity, therefore, is 1M byte. If the bus clock exceeds 33 MHz, the SRAM can be accessed with one wait cycle. At a clock frequency up to 33 MHz, it can be accessed without a wait cycle. The data bus width can be set to 16 or 32 bits, depending on the setting of SW2-4 (BSIZE). Because the high-order bits of the address lines are not decoded, an image appears every 1M byte.

The second-half 32 KB of the SRAM is used by the monitor as a work area, and cannot be used by a user program (see Sections 12.2.3 and 13.2.2).

7.3.3. UV-EPROM (CS0: 0000000 to 03FFFFFF)

A ROM of 128K bytes (64K words \times 16 bits), 256K bytes (128K words \times 16 bits), or 512K bytes (256K words \times 16 bits) with an access time of 120 ns or less can be mounted as UV-EPROM. The type of ROM to be mounted and conditions are specified with SW3 (see Section 5.5). Because the high-order bits of the address lines are not decoded, an image appears for each ROM capacity.

The ROM supports both 32 bits and 16 bits for InitialBusSize. For a 32-bit bus width, the internal ROM is accessed twice because only one 16-bit ROM is used.

Depending on the setting of SW2-3 (BCLK_LOW), the number of ROM wait cycles is changed forcibly by hardware as shown below. If the ROM is accessed with a 32-bit bus width, the number of wait cycles is doubled because the ROM is accessed twice.

SW2-3 (BCLK_LOW): OFF = 8 wait cycles

SW2-3 (BCLK_LOW): ON = 5 wait cycles

7.3.4. Internal ROM (0000000 to 00FFFFFF, 0100000 to 01FFFFFF)

An emulation memory from which data can be fetched at the same access time as the internal ROM is mapped to the internal ROM, allowing the program located in the internal ROM to be executed with its addresses unchanged.

The capacity of the emulation memory is 1M byte. Usually, this memory is read-only, and can be rewritten only with downloading from the debugger.

7.4. I/O MAP

On-board I/O for the RTE-NB85E-CB includes a serial controller (TL16C550C), timer (μ PD71054), LEDs, and switches. Also the chip select space of GBUS is reserved as part of the I/O space. This section explains mapping of the on-board I/O and I/O devices.

7.5. I/O LIST

The table below lists the I/O areas and functions. The number of wait cycles changes depending on the setting of SW2-3 (BCLK_LOW).

Address	Use	Number of wait cycles	
		BCLK_LOW OFF	BCLK_LOW ON
3800000	SW1	10	7
3801000	SW2	10	7
3802000	7SEGLED	10	7
3803000	TOVRDY_LED_CLRPLS	10	7
3804000 to 3804020	PIC	10	7
3807000 to 3807070	UART (TL16C550C)	10	7
3808000 to 3808030	TIC (μ PD71054)	10	7
3809000	SW16	10	7

7.5.1. SW1 Read Port (SW1 3800000H [Read Only])

This port is used to read the status of SW1. The table below indicates the data format.

Physical address	Data bus								Setting
	D7	D6	D5	D4	D3	D2	D1	D0	
3800000H input	SW1 -8	SW1 -7	SW1 -6	SW1 -5	SW1 -4	SW1 -3	SW1 -2	SW1 -1	0 = ON 1 = OFF

SW1-1 corresponds to switch 1 of SW1, while SW1-8 corresponds to switch 8 of SW1. When a bit of the corresponding switch is set to ON, 0 is read. When it is set to OFF, 1 is read. SW1 is used to set the operation of the monitor. For how to set this switch, see Sections 12.1.2 and 13.1.1.

7.5.2. SW2 Read Port (SW2 3801000H [Read Only])

This port is used to read the status of SW2. The data format of this port is shown in the table below.

Physical address	Data bus								Setting
	D7	D6	D5	D4	D3	D2	D1	D0	
3801000H input	SW2 -8	SW2 -7	SW2 -6	SW2 -5	SW2 -4	SW2 -3	SW2 -2	SW2 -1	0 = ON 1 = OFF

SW2-1 corresponds to bit 1 of SW2, and SW2-8 corresponds to bit 8 of SW2. When a bit of the corresponding switch is set to ON, 0 is read; when it is set to OFF, 1 is read. SW2 is used to switch the hardware operation. For the function of each switch, see Section 5.4.

7.5.3. SW16 Read Port (SW16 3809000H [Read Only])

This port is used to read the status of SW16. The data format of this port is shown in the table below.

Physical address	Data bus								Setting
	D7	D6	D5	D4	D3	D2	D1	D0	
3809000H input	SW16 -8	SW16 -7	SW16 -6	SW16 -5	SW16 -4	SW16 -3	SW16 -2	SW16 -1	0 = ON 1 = OFF

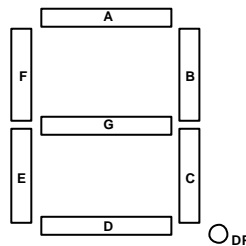
SW16-1 corresponds to bit 1 of SW16, and SW16-8 corresponds to bit 8 of SW16. When a bit of the corresponding switch is set to ON, 0 is read; when it is set to OFF, 1 is read. SW16 is used to switch CPU operation mode. For the function of each switch, see Section 5.7.

7.5.4. 7-Segment LED Display Data Output Port (7SEG-LED 3802000 [Write Only])

This port sets the data to be displayed on the four 7-segment LED. The table below indicates the data format. When a bit is set to 0, the corresponding segment is turned on.

Logical address	Data bus								Setting
	D7..D31	D6..D30	D5..D29	D4..D28	D3..D27	D2..D26	D1..D25	D0..D24	
3802000H output	LED1 -DP	LED1 -G	LED1 -F	LED1 -E	LED1 -D	LED1 -C	LED1 -B	LED1 -A	0 = Turned on 1 = Turned off
3802001H output	LED2 -DP	LED2 -G	LED2 -F	LED2 -E	LED2 -D	LED2 -C	LED2 -B	LED2 -A	
3802002H output	LED3 -DP	LED3 -G	LED3 -F	LED3 -E	LED3 -D	LED3 -C	LED3 -B	LED3 -A	
3802003H output	LED4 -DP	LED4 -G	LED4 -F	LED4 -E	LED4 -D	LED4 -C	LED4 -B	LED4 -A	

The figure below illustrates the correspondence between the bits and the segments of the 7-segment LED.



7.5.5. Time-Over Ready LED Clear Pulse (TOVRDY_LED_CLRPLS 3803000H [Write Only])

If data is written to the port, the TOV_RDY LED, which lights when time-over ready occurs on the board, goes off, and the written data is ignored. Once the TOV_RDY LED is on, it does not go off until data is written to the port or the board is reset.

7.5.6. Interrupt Controller (PIC: 3804000 to 3804020 [Read/Write])

The PIC supports the Multi and PARTNER interrupts necessary for monitor program execution. The interrupts that can be connected are as follows:

- 1) Communication interrupt from RS-232C devices (UART, TL16C550C)
- 2) Timer interrupt request of TOUT0 of the timer (TIC, *n*PD71054)
- 3) Occurrence of time-over ready

Logical address	Register	Data bus							
		D7	D6	D5	D4	D3	D2	D1	D0
3804000H	PIC INT-MASK	x	x	x	x	0	IM2	IM1	IM0
3804010H	PIC INT-STATUS	x	x	x	x	0	IR2	IR1	IR0
3804020H	PIC INTENA	x	x	x	x	0	0	INT0/ NMI-	INT EN

The INT-MASK register masks interrupts applied to its bits. When an INT_MASK bit is set to 1, the interrupt is enabled. When multiple bits are selected, each OR value activates an interrupt.

[Caution] Always write 0 into bit 3 of INT-MASK.

The INTR register is an interrupt status register, for which 1 is read whenever there is an interrupt request. This does not depend on the state of masking. To clear an edge interrupt request, the corresponding bit of this register must be set to 1.

The table below indicates the interrupt source assigned to each bit of IM[0..2] and IR[0..2].

PIC INT-MASK[], STATUS[]	Interrupt source	Request level
0	Timer 0 (mode 2)	Edge (rising)
1	Serial 0	Level (high)
2	Time-over	Level (high)

The INTENA register enables or disables all interrupts.

INTEN: Disables an interrupt by hardware. At this time, the interrupt pin is low.

INTEN	NMI2/INT0
0	Sets a mask. (Reset value)
1	Does not set a mask.

INT0/NMI2-: Selects the interrupt to be used by the monitor.

INT0/NMI-	Interrupt for monitor
0	NMI2 is used. (Reset value)
1	INT0 is used.

INTEN is used to create an edge for an interrupt signal to the CPU each time an interrupt has been serviced when more than one interrupt occurs. If the CPU detects an interrupt using the edge, execute the processing that first sets the INTEN bit to 1 and then clears it to 0 at the last step of the interrupt handler. This allows a pending interrupt to be acknowledged.

[Caution] Do not access the PIC while the monitor is being used.

7.5.7. UART (TL16C550C: 3807000 to 3807070)

The Texas Instruments TL16C550C LSI is used as the UART controller. The TL16C550C has an UART channel. It also has a 16-character FIFO buffer in the transmission/reception block of the UART, and a function for automatically controlling RTS/CTS flow. Therefore, an overrun error of communication can be suppressed by the minimum interrupt.

Each register of the TL16C550C is assigned as listed below. For an explanation of the function of each register, refer to the manual provided with the TL16C550C. (The manual for the TL16C550C is available from the TI&ME of the Texas Instruments home page (<http://www.ti.com/>.)

Address	Read	Write
3807000H	RBR/DLL	THR/DLL
3807010H	IER/DLM	IER/DLM
3807020H	IIR	FCR
3807030H	LCR	LCR
3807040H	MCR	MCR
3807050H	LSR	LSR
3807060H	MSR	MSR
3807070H	SCR	SCR

TL16C550C Register Arrangement

The XIN input of the TL16C550C is connected to the 16-MHz clock.

The UART is connected to the JSIO1 connector of the board. The UART is used to communicate with the host when the remote debugger is used.

TL16C550C is reset when the system is reset.

[Caution] Do not access the UART while the monitor is being used.
--

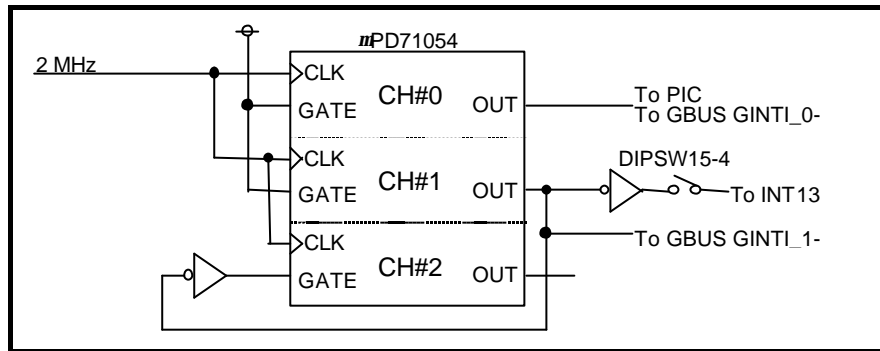
7.5.8. TIC (mPD71054 3808000H to 380803FH)

The NEC mPD71054 is installed as a TIC. The mPD71054 is compatible with the Intel i8254. It has three timers/counters. These timers/counters are used to generate monitor timer interrupts. Each register of the TIC is assigned as listed below.

Address	Read	Write
3808000H	COUNTER#0	COUNTER#0
3808010H	COUNTER#1	COUNTER#1
3808020H	COUNTER#2	COUNTER#2
3808030H	-----	Control Word

TIC Register Arrangement

The channels of the TIC are connected as shown in the figure below. Channel 0 is used as the interval timer for the Multi ROM monitor program. Channels 1 and 2 can be used by a user program as necessary. Channel 2 is connected to channel 1 by means of a cascade connection.



Examples of modes

- CH#0: Mode 2 (rate generator)
- CH#1: Mode 2 (rate generator)
- CH#2: Mode 0 (down counter)

[Caution] Do not access CH0 while the monitor is being accessed.

8. SOFTWARE

This chapter describes the initialization of the hardware of the RTE-NB85E-CB board, and explains how to use peripheral devices.

8.1. INITIALIZATION

To write a program that is booted from ROM without using the monitor, initialize the internal bus controller of the NB85E in the first routine of the program. For the values to be set for initialization, see Section 7.2

8.2. SUCCESSIVE ACCESSES TO *m*PD71054

To successively access the *m*PD71054, access other spaces at least once between the first and second accesses to the *m*PD71054. These accesses ensure the recovery time of the *m*PD71054.

Recovery time is also assured by a dummy read of a resource (such as ROM) other than the *m*PD71054.

8.3. LIBRARIES

Libraries are required for programming using the C compiler for I/O accesses and other purposes. However, the methods of writing these libraries and passing their parameters described below are specific to the Multi. So, modifications may be required, for example, when another compiler is used.

```

/* I/O library */

/* GHS V850 compiler parameter passing */
/* arg0:r6, arg1:r7, arg2:r8, return:r10 */

inb(int addr)                /* Byte (8 bits) input */
{
    __asm(" ld.b 0[r6], r10");
}

inh(int addr)                /* Half-word (16 bits) input */
{
    __asm(" ld.h 0[r6], r10");
}

inw(int addr)                /* Word (32 bits) input */
{
    __asm(" ld.w 0[r6], r10");
}

outb(int addr, int data)     /* Byte (8 bits) output */
{
    __asm(" st.b r7, 0[r6]");
}

outh(int addr, int data)     /* Half-word (16 bits) output */
{
    __asm(" st.h r7, 0[r6]");
}

outw(int addr, int data)     /* Word (32 bits) output */
{
    __asm(" st.w r7, 0[r6]");
}

```

8.4. EXAMPLE OF USING TIMERS

A sample time measurement is indicated below which uses timer 1 and timer 2 cascaded with each other by an external timer (μ PD71054) on the board. Timer 1 is initialized as an interval counter (mode 2), and timer 2 is initialized as a down counter (mode 0). By determining the counter values before and after a routine whose execution time is to be measured, the execution time can be calculated. Note that both timers function as down counters. Note also that command recovery (for example, SW1 dummy read) is required for successive accesses to the external timer.

```

/* Sample execution time measurement using timers */

#define TIMERCLK      2000000          /* 2 MHz */
#define INTERVAL     (TIMERCLK * 10 / 1000) /* 10 ms (1/100) */
#define IOWAIT()     (*(char *) 0x3800000) /* For I/O command recovery */

InitTimer() /* Timer initialization */
{
    outb(0x3808030, 0x74);          IOWAIT(); /* Timer 1 set to mode 2 */
    outb(0x3808010, INTERVAL);     IOWAIT(); /* Lower-digit count of timer 1 */
    outb(0x3808010, INTERVAL / 256); IOWAIT(); /* Higher-digit count of timer 1 */
    outb(0x3808030, 0xB0);          IOWAIT(); /* Timer 2 set to mode 0 */
    outb(0x3808020, 0xFF);          IOWAIT(); /* Lower-digit count of timer 2 */
    outb(0x3808020, 0xFF);          IOWAIT(); /* Higher-digit count of timer 2 */
    return 0;
}

LatchTimer() /* Count latch */
{
    int count1, count2, counts;

    outb(0x3808030, 0xDC);          IOWAIT(); /* Timer 1/2 multiple latch */
    count1 = inb(0x3808010);         IOWAIT();
    count1 += inb(0x3808010) * 256;  IOWAIT(); /* Count of timer 1 */
    count2 = inb(0x3808020);         IOWAIT();
    count2 += inb(0x3808020) * 256;  IOWAIT(); /* Count of timer 2 */
    counts = INTERVAL * (0xFFFF - count2)
            + (INTERVAL - count1);
    return counts;
}

double total_time;

main()
{
    int start_count, stop_count;

    InitTimer();
    start_count = LatchTimer();      /* Start count value */
    func();
    stop_count = LatchTimer();       /* Stop count value */
    total_time = (double)(stop_count - start_count)
                / (double)TIMERCLK; /* Seconds */
    return 0;
}

#include <time.h>

func() /* Time measurement routine */
{
    ....
}

```

9. DEVELOPMENT OF APPLICATIONS USING MASKABLE INTERRUPTS

This chapter describes the methods of developing an application on the RTE-NB85E-CB by using a maskable interrupt, and related restrictions.

9.1. INTERRUPT VECTOR

The NB85E interrupt vector area of addresses 0000000H to 00007FFH is fixed in the ROM in Singlemode1 and RomlessMode, and cannot be rewritten. So, for the monitor, the following two vector areas are allocated in the SRAM:

Alternate vector area:

This vector area can be rewritten by the user program. It is used if a relative jump can be executed from the interrupt vector area. The branch instruction for the relative jump is placed in the vector area in this case.

Relay vector area:

This vector area is used by the monitor if a relative jump from the interrupt vector area cannot be executed. In this case, an instruction that saves the registers and a branch instruction for an absolute jump are placed in the interrupt vector area, and an instruction that restores the registers and a branch instruction for a relative jump to the alternate area are placed in this area.

The following table shows the alternate vector area.

Mode	Alternate vector area	Relay vector area
SINGLE MODE0	3CF8000H to 3CF87FFH	3CF8800H to 3CF8FFFH
SINGLE MODE1	3CF8000H to 3CF87FFH	3CF8800H to 3CF8FFFH
ROMLESS	3CF8000H to 3CF87FFH	3CF8800H to 3CF8FFFH

* The relay vector area can be used by the user program.

If, for example, an interrupt with exception code 0080H is generated, the CPU interrupt function causes a branch to address 0000080H, where an instruction for causing a branch to 0080H (the offset of alternate vector area) is placed. By rewriting the alternate vector area in the destination, a branch to the user program interrupt handling routine can be caused when an interrupt is generated.

If an exception code 0080H interrupt occurs, therefore, write an instruction that branches execution to the specified interrupt processing at address 3CF8080H of the alternate vector area in SRAM.

The difference from an ordinary NB85E program is that a vector area is fixed in ROM, and no setting (rewriting) by a program is required. However, a program using the monitor on the RTE-NB85E-CB must rewrite the vector area by the program to enable an interrupt.

A sample program for alternate vector rewriting is given below (when the relative address from the interrupt handling routine to an alternate vector area is within 22 bits).

```

#define IROM_WRENA    1
#define IROM_WRDIS    0

void di() /* Disable interrupt */
{
    __asm( "di" );
}
void ei() /* Enable interrupt */
{
    __asm( "ei" );
}
void SetAJump(int addr, int jmpdest) /* ← Vector setting routine */
/* int addr; address where we're storing the 'jr' */
/* int jmpdest; address where the 'jr' jumps to */
{
    int offset;
    unsigned inst;
    unsigned int *p ;

    offset = jmpdest - addr;
    inst = 0x07800000 /* 'jr' opcode */ | (offset & 0x003ffffe);
    *((UINT16 *) (addr + 0)) = (inst >> 16) & 0xffff ;
    *((UINT16 *) (addr + 2)) = (inst & 0xffff) ;
}
.....
void __interrupt IntEntry() /* ← Interrupt handling routine */
{
    .....
}
.....
main()
{
    .....
    SetAJump((int)(0x080 + 0x3CF8000) ,(int)IntEntry) ;
    /*          ↑ Exception code of specified interrupt */
}

```

9.2. GENERAL RESTRICTIONS/NOTES

This section describes restrictions and notes relating to the debugging of an application using a maskable interrupt.

- 1) If an interrupt is generated before alternate vector setting, or if an interrupt is generated with other than a valid alternate vector set, a break occurs at the point where the interrupt is generated. This is because the initial value of the alternate vector is an instruction for causing a branch to the break handling routine of the monitor.
- 2) If the relative address from an alternate vector area to the interrupt handling routine exceeds 22 bits, the contents of at least one register must be destroyed, or a branch relay point must be created, to cause a branch to the interrupt handling routine.
- 3) The alternate vector area and the vector area for SingleMode0 mode can be rewritten when the program is downloaded (see Section 9.3). To rewrite these areas when the program is downloaded, however, rewrite only the interrupts that are used.
- 4) All peripherals, including interrupt-related peripherals, can be initialized only with the reset switch on the board. This means that if, after a program is executed, another program is loaded, the peripherals will still be in the statuses set by the previous program. So, use the procedure below when, for a program that uses a peripheral, another program is to be loaded and executed.
 - (1) Disconnect the monitor.
 - (2) For resetting the board, press the reset switch of the RTE-NB85E-CB.
 - (3) Connect the monitor.
 - (4) Load and execute another program.
- 5) Before setting the EI (interrupt enable) state, set the DI (interrupt disable) state at the start of program execution, then set the peripherals and vectors.
- 6) To disable (DI) or enable (EI) an interrupt during a break with the I/O (register) manipulation function of the debugger, use the corresponding bit of the interrupt mask register (IMRn). If the interrupt control register (PICn or PnnICn) is manipulated with the I/O (register) manipulation function of the debugger during a break, do nothing to the interrupt control register, since the interrupt operation may not be performed correctly.

9.3. REWRITING THE ALTERNATE VECTOR AREA DURING DOWNLOADING

A vector can be rewritten in various ways while a program is being downloaded. This section shows an example in the Multi environment of GHS. The method used can be thought of as being similar to a program stored to ROM. Also see the program example shown above.

- 1) Defining program for rewriting the interrupt vector area (ASM language)

Define a program consisting of only the branch instructions to be placed in the interrupt vector as shown below. For details of the coding, see the manual for the language processor.

```

.section      "intvct", .text      /* Defined section name */
.align      4
.globl      _Int80
_Int80:
    jr _IntEntry                /* jump to handler */
    nop
    nop

```

Note, however, that the program cannot be defined in such a way that it exceeds the vector boundary for one interrupt.

2) Defining the section map

Define the section map that is used during linking as follows. For details of the coding, see the manual for the language processor.

```
{
    .intvct      0x3FE8080  :
    .
    .text       0x3E00000  :
    .data       align(0x10) :
    .
    .
    .
}
```

First define the section of the program to be placed in the vector.

To use more than one interrupt, define one section if the vectors are contiguous (the interrupt vector boundaries must match). If the vectors are not contiguous, define a section for each interrupt, and specify all the sections in the section map.

In this way, a specific location for the alternate vector area can be rewritten when the program is downloaded, and the interrupt vector can be easily rewritten even by a program that is executed in the internal ROM area. In addition, no code is necessary to rewrite the interrupt vector.

9.4. RESTRICTIONS/NOTES ON BREAKPOINTS

Note that the following restrictions and notes must be observed when a breakpoint is set or executed (single step) in an interrupt handling routine.

- 1) During a break, all maskable interrupts are rejected.
- 2) The single step function sets a temporary breakpoint in the next instruction. As a result, even though the user program in the EI (enable interrupt) status is executed on a single-step basis, execution can branch to the interrupt handler for handling of an interrupt while one instruction is being executed. Be sure, therefore, to observe the points noted regarding breakpoints during single-step execution.
- 3) Exiting from the interrupt handling routine by single stepping is impossible. (Specifically, single stepping based on the last "}" of the interrupt handling routine is disabled.) Similarly, reti instruction single stepping is impossible. The Return function of the debugger does not support return from an interrupt handling routine to the original routine.

10. CPU PIN CONNECTION

This chapter explains the uses of the CPU pins in the RTE-NB85E-CB.

10.1. LIST

The table below lists the uses of the main CPU pins. Details are given in the subsequent sections.

Pin name	Use	Section reference
D0 to D31	Used as the system data bus.	
A0 to A25	Used as the system address bus.	
BCYST- RD-	Used as the system bus control signals.	
WR0-/CAS0-/DQM0 to WR3-/CAS3- /DQM03	Used by the system bus and SDRAM.	
OE- WE- SDRAS- SDCAS- CKE SDCLK CS3-/RAS3-	Used by SDRAM.	
CS0-/RAS0- CS1-/RAS1- CS2-/RAS2- CS4-/RAS4- CS5-/RAS5- CS7-/RAS7-	Used by system bus as CSx-.	
CS6-/RAS6-	Not used.	
HLDREQ-	Not used.	
HLDAK-	Not used.	
VRESZ	Inputs RESET.	10.2
MWAIT-	Used by system bus. Hi-Z and pulled down to 1 kΩ in the CS6 space.	10.3
NMIO to NMI2, INTO	Used by the system (can be disconnected with SW11).	10.4
INT10 to INT13	Used by the system (can be disconnected with SW15).	10.5
RXD/IN45	Used as SIO2-RXD (can be disconnected with SW14).	10.6
TXD/INT46	Used as SIO2-TXD.	10.7
PORT0/MPXSCZ PORT1/DSTBZ PORT2/RDCYZ PORT3/BUSSR	Used as PORT to control SIO2 (can be disconnected with SW12).	10.8
DMARQ0/INT32 DMARQ1/INT33 DMARQ2/INT34 DMARQ3/INT35	Used as DMARQ to connect GBUS-DMARQ0-3 (can be disconnected with SW13).	10.9
DMAAK0/INT36 DMAAK1/INT37 DMAAK2/INT38 DMAAK3/INT39	Used as DMAAK and connected to GBUS-DMAAK0-3 (can be disconnected with SW13).	10.10

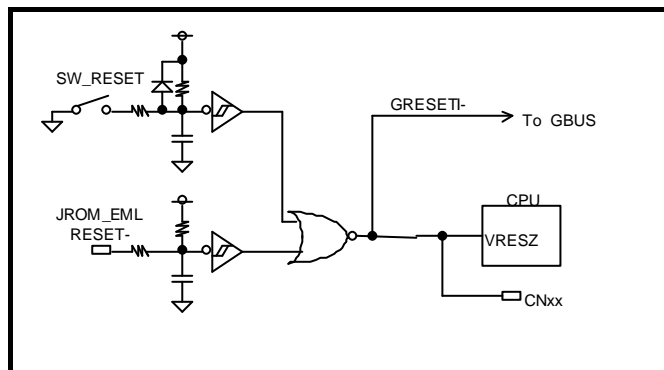
Pin name	Use	Section reference
TC0/INT40 TC1/INT41 TC2/INT42 TC3/INT43	Connected to reserve pin of GBUS (can be disconnected with SW14).	10.11
TB136, TB137 STOPZ	Pulled up to 10 kΩ.	10.12
INTxx VB0 to VB31 VPD0 to VPD14 VPRETR, VPDACT TBI32 to TBI34, TBI38, TBI39 TEST, BUNRI VAREQ0, VBWAIT, VBHOLD, VBLAST VBEXDC, VBEXCLK,	Pulled down to 10 kΩ.	10.12
Others	Not connected.	

10.2. RESET-

The factors listed below trigger a CPU reset. These factors reset the CPU. They also system -reset the board.

- Power-on reset: Occurs when the power to the board is switched on.
- Reset request received from the JROMEM connector: Reset by input from the RESET- pin of the JROMEM connector. (See Section 5.10.)
- Reset by the SW_RESET: Generated by the reset switch (SW_RESET) on the rear panel. (See Section 5.1.)

The figure below outlines the reset signal generation logic.



10.3. MWAIT-

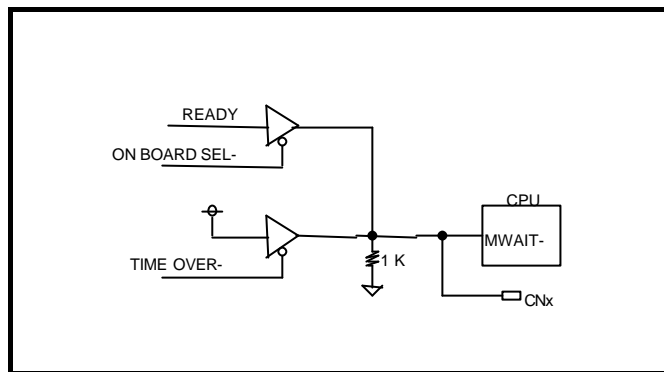
The MWAIT pin drives a ready signal if a resource in the board is accessed and if time-over ready occurs.

Time-over ready is a function that, if a bus cycle is not completed within a specific time for some reason, detects the lack of completion and completes the bus cycle forcibly.

If time-over ready occurs, TOVER_LED on the board lights and an interrupt is issued to the PIC. TOVER_LED remains lit until a time-over ready LED clear pulse is generated by software or the board is reset (see Section 7.5.5).

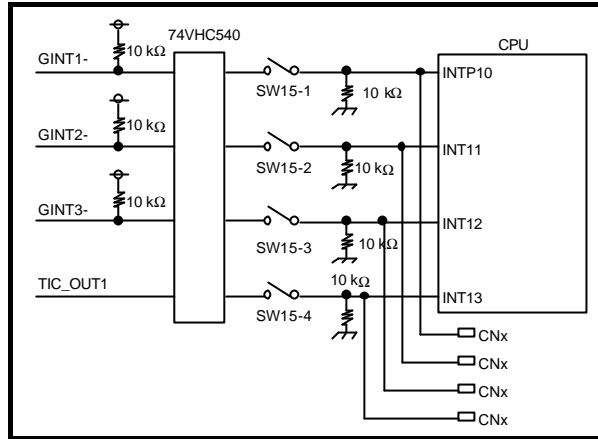
For access to the space allocated to GBUS, time-over ready occurs only when the GMOTHER_DETECT- signal of GBUS is high (when the board is not connected to GBUS).

The configuration of the MWAIT drive block is shown below.



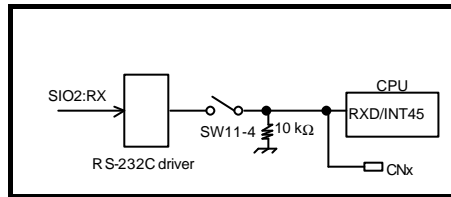
10.5. INT10 TO INT13

INT10 to INT13 are connected to GBUS-INT0, 1, 2, and TIC_OUT1 (output of timer CH1) via a switch. Connection of each pin is shown below.



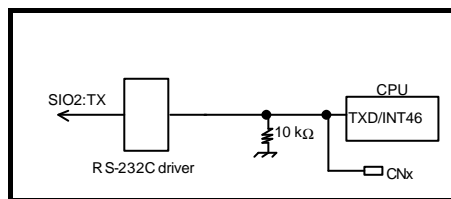
10.6. RXD/INT45

A signal that is Rx of SIO2 converted to TTL level by an RS-232C receiver driver is connected to the RXD/INT45 pin via a switch, as shown below.



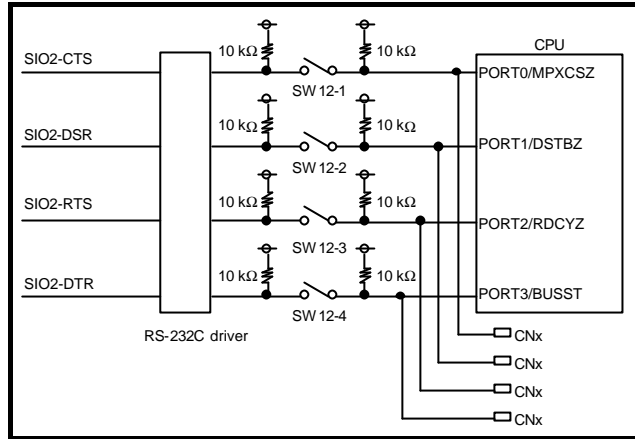
10.7. TXD/INT46

The TXD/INT46 pin is used for TX of SIO2 via an RS-232C transmitter driver, as shown below.



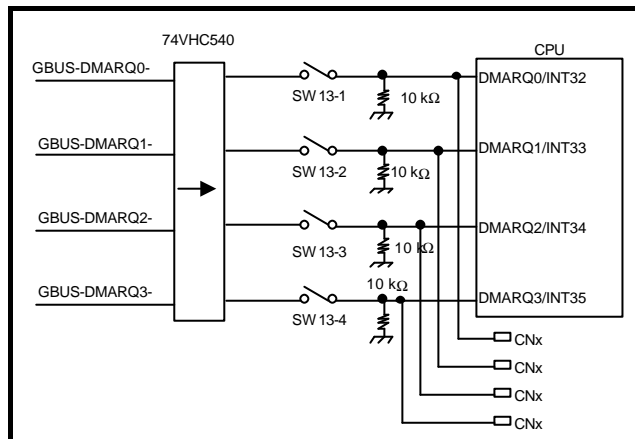
10.8. PORT0/MPXCSZ, PORT1/DSTBZ, PORT2/RDCYZ, PORT3/BUSST

These pins are used for CTS, DTR, RTS, and DTR of SIO2 via an RS-232 transmitter/receiver driver, as shown below.



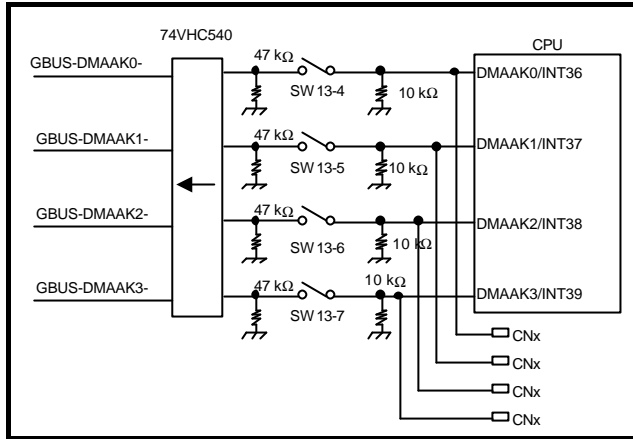
10.9. DMARQ0/INT32, DMARQ1/INT33, DMARQ2/INT34, DMARQ3/INT35

The logic of a DMARQ request from GBUS is inverted and connected to these signal pins via a switch, as shown below.



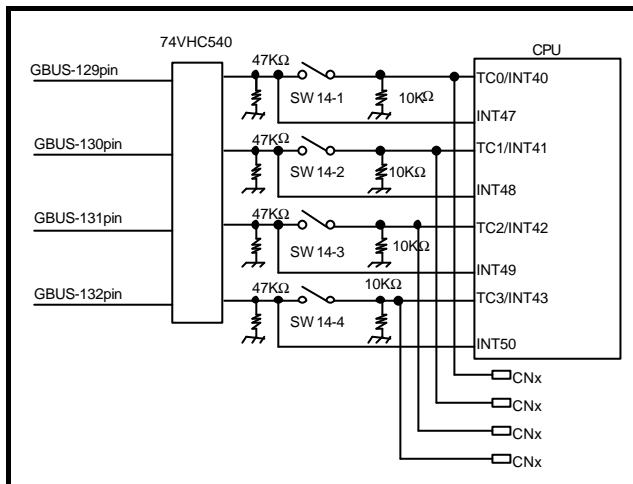
10.10. DMAAK0/INT36, DMAAK1/INT37, DMAAK2/INT38, DMAAK3/INT39

These signal pins invert the logic of a signal output by the CPU and are connected to DMAAK of GBUS via a switch, as shown below.



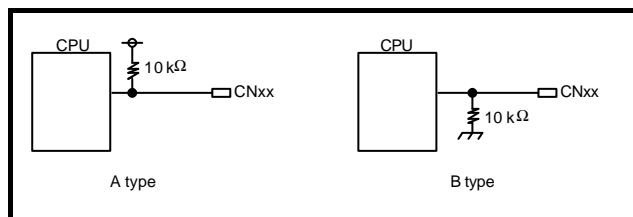
10.11. TC0/INT40, TC1/INT41, TC2/INT42, TC3/INT43

These signal pins invert the logic of a signal output by the CPU and are connected to a reserve pin of GBUS via a switch, as shown below.



10.12. OTHER SIGNALS

Board signals not used are connected to the JCPU connector as shown below.



11. SPECIFIC GBUS SPECIFICATIONS

This chapter explains how GBUS is used with the RTE-NB85E-CB. For the general GBUS specifications, see Chapter 14.

11.1. GENERAL

The following table shows the GBUS signal lines used by the RTE-NB85E-CB.

GBUS signal name	Function	See
GADDR[31:2]	Used as address lines. GADDR[26:31] are not connected. GADDR[25:24] are don't care.	
GDATA[31:0]	Used as data lines. In a read cycle, the signal that is latched on the rising edge of VBCLK is supplied to the CPU.	
GCS-[6:0]	Used as chip select lines.	
GCLK	An asynchronous 33-MHz clock is connected to VBCLKT of the CPU.	
GRESETI-	Outputs the reset request generated by the board.	
GRESETO-	Not connected	
GADS-, GREADY-, GBLAST-, GW/R-	Used as bus control signals.	
GWAITI-	Not connected	
GBTERM-	Not connected	
GRD-, GWR-	RD- and WR- signals generated from the GBUS control signals are connected.	
GHOLD-, GHLDA-	Not connected	
GBREQ-	Not connected	
GDMARQ-[3:0]	Used as DMA request signal.	10.9
GDMAAK-[3:0]	Used as DMA acknowledge signal.	10.10
GINTO-[3:0]	Used as interrupt request signal.	10.5
GINTI-[1:0]	OUT0 and OUT1 of TIC (μ PD71054) are connected to GINTI0- and GINTI1-.	
GETC[7:0]	Not connected	
GAHI_EN-	Not connected	
GMOTHER_DETECT-	Used by time-over ready generation circuit.	
GUSE_DIRECT_ACC-	Not connected	
GCLK_LOW-	Not connected	
GLOCK-[1:0]	Not connected	

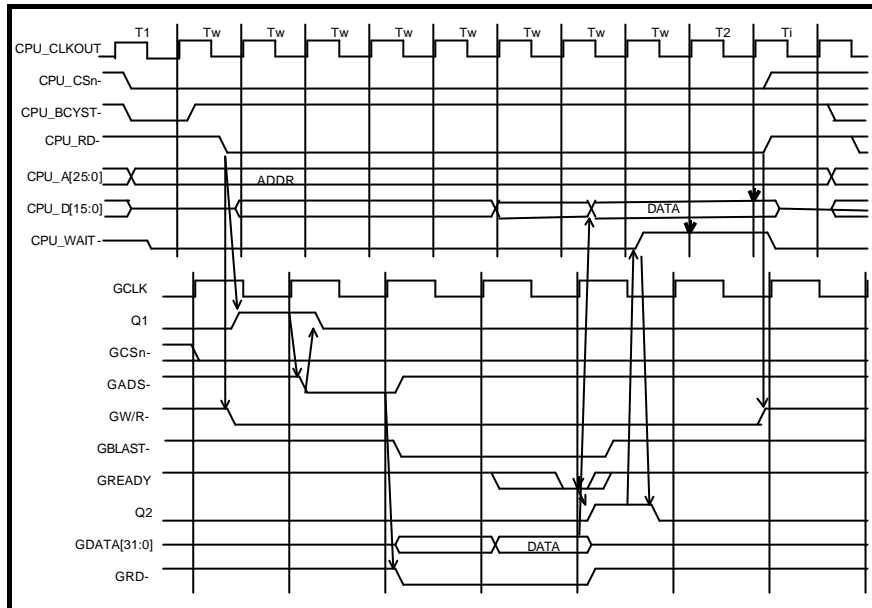
11.2. BUS CYCLE

A 33-MHz clock that is asynchronous with CLK of the CPU is connected to GCLK of GBUS. In addition, because GAHI_EN- is not connected, GADDR[26:31] are not connected. GADDR[24:25] are always [0, 0].

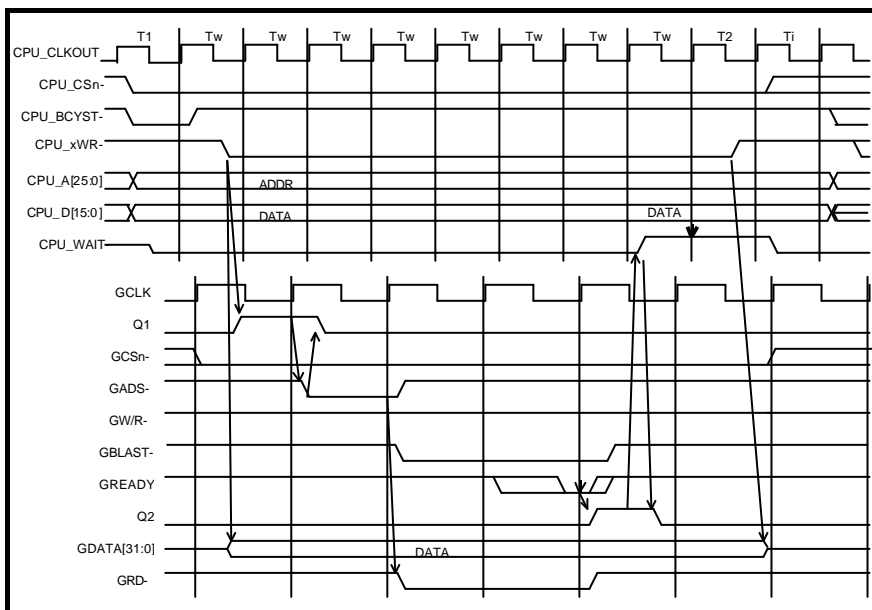
A read cycle from GBUS can be executed on GBUS without a wait cycle.

The CPU_xxx signal shown in the timing charts below is a CPU signal. The Gxxx signal is a GBUS signal.

The following chart shows a read cycle.



The following chart shows a write cycle.



11.3. CHIP SELECT

On the board, the following spaces are allocated to GBUS chip select. For how to set the internal bus configuration register of the CPU for the following spaces, see Section 7.2.1.

GBUS signal	CPU address space	Physical address range	RTE-MB-A resources
GCS0-	Entire CS2 space	0400000 to 07FFFFFF	Common SRAM (2M)
GCS1-	Space of ADDR[23] = 0 of CS4 space and entire CS0 space if SW2-1 (FBOOT) is ON	2000000 to 27FFFFFF 0000000 to 03FFFFFF	Flash ROM (8M)
GCS2-	Space of ADDR[21..19] = [010] of CS5 space	3900000 to 397FFFF	I/O register
GCS3-	Space of ADDR[23] = 1 of CS4 space	2800000 to 2FFFFFF	EXT-bus: memory space
GCS4-	Space of ADR[21] = 1 of CS5 space	3A00000 to 3BFFFFFF	EXT-Bus: I/O space
GCS5-	Entire CS1 space	0800000 to 0FFFFFF	PCI bus space
GCS6-	Space of ADDR[21..19] = [011] of CS5 space	3980000 to 398FFFF	PCI-Cont register

12. APPENDIX A Multi MONITOR

This chapter describes how to make the settings required to establish a connection between the Multi monitor stored in ROM and the Multi debugger on the host. It also provides notes on the use of the Multi monitor.

12.1. BOARD SETTING

12.1.1. RTE for Win 32 Installation

When the board is used with the Multi debugger, communication software called RTE for Win32 must be installed in the PC. Refer to the RTE for Win32 Installation Manual (supplied with this product) for installation and test methods.

12.1.2. SW1 Setting

SW1 is a switch for general-purpose input ports. For the Multi monitor in the factory-installed ROM, SW1 is used as shown below.

SW1	1	2	Baud rate
Setting	ON	ON	115,200 baud
	OFF	ON	38,400 baud
	ON	OFF	19,200 baud
	OFF	OFF	9,600 baud (Factory-set)

Baud Rate Setting

SW1	3	4	Profiler period
Setting	ON	ON	Timer is not used.
	OFF	ON	200 Hz 5 ms
	ON	OFF	100 Hz 10 ms
	OFF	OFF	60 Hz 16.67 ms (Factory-set)

Profiler Period Setting

SW1	8	Debug mode
Setting	ON	7-segment LED used by the monitor
	OFF	Normal use state (Factory-set)

Debug Mode Setting

SW1-5 to SW1-7 are not used with the Multi monitor.

12.1.3. Setting of Other Switches

The monitor changes its initialization or operation depending on the setting of SW2 and SW3, and the CPU operation mode (see Sections 5.4, 5.5, and 5.7).

12.1.4. Connection of Board

Connect the board to the PC serially, by referring to Chapter 6.

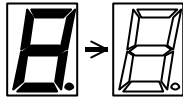
12.2. Multi MONITOR

12.2.1. 7-Segment LED on Startup

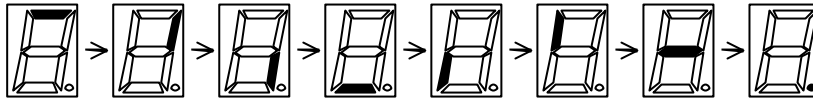
The 7-segment LED of the ROM monitor for Multi operates as follows when power is supplied to the board (black indicates the segment that lights).

- 1) Check operation of 7-segment LED (See figure below.)

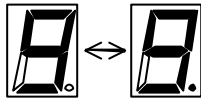
If SW1-8 is OFF:



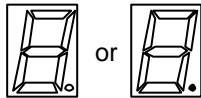
If SW1-8 is ON:



- 2) Number counting by simple SRAM memory check
* Not executed if SW1-8 is OFF
- 3) Connection wait status (The dot does not blink if the profiler timer is stopped.)



- 4) Connection status (The status of the dot is retained on connection.)



12.2.2. ROM Monitor Work RAM

The ROM monitor uses the first 32K-byte area (3FE8000H to 3FEFFFFH) in the SRAM as work RAM. In other words, user programs are not allowed to use this area or its image area.

12.2.3. Monitor Interrupt

The interrupt selected by SW2-5 is used for monitor communication, the timer, and forced break.

12.2.4. _INIT_SP Setting

_INIT_SP (stack pointer initial value) is set to 3FE7FF0H (immediately before monitor work RAM) by the monitor. (_INIT_SP can be changed in the Multi debugger environment.) The monitor uses a 32-byte stack area set by the user program.

12.2.5. Timer Interrupt

If the timer interrupt is disabled, the profiler function of Multi cannot be used (for how to set the timer interrupt, see Section 12.1.2).

12.2.6. Initializing Hardware

The ROM monitor performs initialization so that the resources on the board can be directly accessed.

12.2.7. Special Instruction

The monitor uses the following instruction for the single step, breakpoint, and system call functions.

BRKTRAP instruction (0xnn40)

Do not use a code that may be interpreted as a break instruction in the user program.

12.3. RTE COMMANDS

When the monitor and server are connected, the TARGET window is opened. The RTE commands can be issued in this window. The following table lists the RTE commands.

Command	Description
HELP, ?	Displays help messages.
INIT	Initializes.
VER	Displays the version number.
SFR	Displays or sets the internal I/O.

RTE Commands

Some commands require parameters. All numeric parameters such as addresses and data are assumed to be hexadecimal numbers. The following numeric representations are invalid:

0x1234 1234H \$1234

12.3.1. HELP(?)

<Format> HELP [command-name]

Displays a list of RTE commands and their formats. A question mark (?) can also be used in place of the character string HELP. If no command name is specified in the parameter part, the HELP command lists all usable commands.

<Example> HELP SFR

Displays help messages for the SFR command.

12.3.2. INIT

<Format> INIT

Initializes the RTE environment. Usually, this command should not be used.

12.3.3. VER

<Format> VER

Displays the version number of the current RTE environment.

12.3.4. SFR Command

<Format> SFR [register-name [=data]]

When a register name is specified with data omitted, the data read from the register is displayed. When a register name is specified, and data is specified after =, the data is written to the register. The size of data is automatically determined according to the valid size of the specified register. For details of the internal I/O registers, refer to the manual provided with the NB85E CPU.

<Example 1> SFR

A list of registers is displayed.

<Example 2> SFR IMR

The contents of the IMR register are displayed.

<Example 3> SFR IMR=55AA

Data 55AAH is written into the IMR register.

13. APPENDIX B PARTNER MONITOR

This chapter describes how to make the settings required to establish a connection between the PARTNER monitor stored in ROM and the PARTNER on the host. It also provides notes on the use of the PARTNER monitor.

13.1. BOARD SETTING

13.1.1. SW1 Setting

SW1 is a switch for general-purpose input ports. For the PARTNER monitor in the factory-installed ROM, SW1 is used as shown below.

SW1	1	2	Baud rate
Setting	ON	ON	115,200 baud
	OFF	ON	38,400 baud
	ON	OFF	19,200 baud
	OFF	OFF	9,600 baud (Factory-set)

Baud Rate Setting

SW1	3	4	Timer
Setting	ON	ON	Always use this switch in this status. (Factory-set)

SW1	8	Debug mode
Setting	ON	7-segment LED used by the monitor
	OFF	Normal use state (Factory-set)

Debug Mode Setting

SW1-5 to SW1-7 are not used with the PARTNER monitor.

13.1.2. Setting of Other Switches

The monitor changes its initialization or operation depending on the setting of SW2 and SW3, and the CPU operation mode (see Sections 5.4, 5.5, and 5.7).

13.1.3. Connection of Board

Connect the board to the PC serially, by referring to Chapter 6.

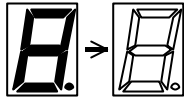
13.2. PARTNER MONITOR

13.2.1. 7-Segment LED on Startup

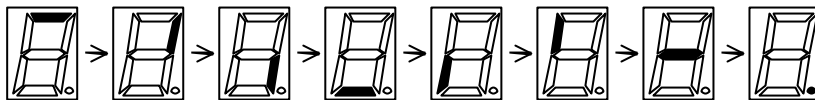
The 7-segment LED of the ROM monitor for PARTNER operates as follows when power is supplied to the board (black indicates the segment that lights).

- 1) Check operation of 7-segment LED (See figure below.)

If SW1-8 is OFF:



If SW1-8 is ON:



- 2) Number counting by simple RAM memory check

* Not executed if SW1-8 is OFF

- 3) Connection wait status



- 4) Connection status



13.2.2. ROM Monitor Work RAM

The ROM monitor uses the first 32K-byte area (3FE8000 to 3FEFFFF) in the SRAM as work RAM. In other words, user programs are not allowed to use this area or its image area.

13.2.3. Monitor Interrupt

The interrupt selected by SW2-5 is used for monitor communication and forced break (ESC button).

13.2.4. SP Setting

The stack pointer initial value is set to 3FE-7FF0H (immediately before monitor work RAM) by the monitor. The monitor uses a 32-byte stack area set by the user program.

13.2.5. Initializing Hardware

The ROM monitor performs initialization so that the resources on the board can be directly accessed.

13.2.6. Special Instruction

The monitor uses the following instruction for the single step, breakpoint, and system call functions.

BRKTRAP instruction (0xnn40)

Do not use a code that may be interpreted as a break instruction in the user program.

14. APPENDIX C GBUS COMMON SPECIFICATIONS

This appendix explains the GBUS specifications that are not dependent on the type of board.

14.1. TERMINOLOGY

Terminology used in this appendix is explained below.

14.1.1. CPU Board and Motherboard

A board in the RTE-CB series is called a CPU board and a Midas lab board connected to GBUS of the CPU board is called a motherboard.

14.1.2. Bus Cycle and Micro Cycle

GBUS is a general bus that can be accessed in burst mode.

A bus cycle consists of a series of cycles, including a one in which a burst access occurs, that is completed (asserting of GADS- is necessary to mark the end of a bus cycle).

Bus cycles are classified into single cycles and a burst cycles. A single cycle is a bus cycle in which data transfer occurs only once. A burst cycle is a bus cycle in which data transfer occurs two or more times.

One cycle for each data transfer in a burst cycle is called a micro cycle.

14.2. SIGNALS

The GBUS signals are listed below. The input/output direction of each GBUS signal is indicated as viewed from the motherboard. Therefore, "input" means that a signal output from the CPU board is input to the motherboard (this also applies to signal names).

"Bidirectional" signals change direction depending on the status of the bus cycle.

"Input/output" signals also change direction depending on whether the bus master is the CPU board or motherboard. The direction written first is the signal direction when the CPU board is the bus master, and the direction written later is the signal direction when the motherboard is the bus master.

A GBUS signal is a +5-V TTL level signal. The motherboard is always little endian.

Signal name	Input/output	Function
GCLK	Input	<ul style="list-style-type: none"> • Synchronization clock of GBUS. The maximum frequency is 33.33 MHz, and the minimum frequency is 10.0 MHz. GBUS operates synchronized with the rising edge of this clock. • Since, on the motherboard, this clock is terminated at 330 Ω with respect to +5V and GND, the circuit on the CPU board must be able to drive this resistance. • If GCLK is less than 16.67 MHz, GCLK_LOW- goes low. In this way, the motherboard can adjust the number of wait cycles. • Because a PLL (Phase Lock Loop) zero delay buffer may be used, if the frequency of GCLK is changed, the motherboard must not be accessed for at least 1 ms after the frequency has been changed to allow the PLL to be locked.
GRESETI-	Input	<ul style="list-style-type: none"> • Reset signal of GBUS. If a reset occurs on the CPU board, this signal goes low. The motherboard is reset by this signal (the motherboard can also be reset for other causes on the motherboard).
GRESETO-	Output	<ul style="list-style-type: none"> • This signal goes low if the motherboard is reset. • The motherboard ORs the reset signal on the motherboard with GRESETI- as GRESETO-. Accordingly, the CPU board resets the circuits on the CPU board by ORing GRESETI- and GRESETO- (GRESETI- and GRESETO- are ORed because there is a possibility that the motherboard is not connected).

Signal name	Input/output	Function
GADDR[31:2]	Input/output	<ul style="list-style-type: none"> Address signals of GBUS. These signals are driven by a valid value during a cycle. GADDR[31] is ignored on the motherboard if the CPU is the bus master. The low-order addresses A1 and A0 use a byte enable signal. GADDR[31:26] from the CPU board can be treated as 0 by using the GAHI_EN- signal. If the bus master is the motherboard and if GADDR[25] is 0, the resources on the motherboard are selected; if GADDR[25] is 1, the resources on the CPU board is selected.
GBEN-[3:0]	Input/output	<ul style="list-style-type: none"> Byte enable signals of GBUS. These signals are always driven by a valid value during a cycle. GBEN0-, GBEN1-, GBEN2-, and GBEN3- correspond to byte lanes GDATA[7:0], GDATA[15:8], GDATA[23:16], and GDATA[31:24], and the corresponding byte lane is valid if GBENx- is low.
GDATA[31:0]	Bidirectional	<ul style="list-style-type: none"> Bus data signals of GBUS. These signals are pulled up to 10 kΩ on the motherboard. The direction of these signals is determined by GW/R-.
GADS-	Input/output	<ul style="list-style-type: none"> Address strobe signal of GBUS. If this signal is sampled low on the rising edge of GCLK, the start of a bus cycle is indicated. The motherboard ignores GADS- if none of the chip select signals (GCS-[7:0]) is active.
GREADY -	Output/input	<ul style="list-style-type: none"> Ready signal of GBUS. If this signal is sampled low and GWAITI is sampled high on the rising edge of GCLK during a micro cycle, the end of the micro cycle is indicated. Time-over ready when the CPU board accesses the motherboard is generated by the motherboard. The reason is to avoid collision with the GREADY - signal.
GWAITI-	Input	<ul style="list-style-type: none"> Wait request signal. This signal is sampled on the rising edge of GCLK. If the CPU board cannot support a cycle with a few wait cycles, the CPU board samples GWAITI- low at the sample timing of GREADY - so that the motherboard can handle GREADY - as a ready signal even though it is low at the time. Usually, this signal is used if the CPU board cannot support zero wait burst (see Section 14.6.3). This signal is valid only in a cycle in which the CPU board is the bus master.
GBLAST-	Input/output	<ul style="list-style-type: none"> Bus cycle completion notification signal. This signal is sampled on the rising edge of GCLK. This signal is asserted low by the bus master when a micro cycle that completes the bus cycle starts. The bus cycle is completed if the low level of GBLAST-, low level of GREADY -, and high level of GWAITI- are sampled on the rising edge of GCLK.
GBTERM-	Output/input	<ul style="list-style-type: none"> Bus cycle completion request signal. This signal is sampled on the rising edge of GCLK. If the accessed side requests completion of the bus cycle, the GREADY - and GBTERM- signals go low. If the bus master samples GBTERM- as low when it samples GREADY - as low, it must complete the bus cycle even though GBLAST- has not been asserted, and start the bus cycle again by asserting GADS- again. GBTERM- must be asserted at the same time as GREADY -. This signal is used to complete the bus cycle if the accessed side does not support burst cycles or if a burst cycle exceeding the supported number of bursts is requested.

Signal name	Input/output	Function
GW/R-	Input/output	<ul style="list-style-type: none"> • Write/Read signal. This signal indicates the direction of the data bus. It is always driven by a valid value during the bus cycle. • This signal indicates the direction of the data bus for the bus master.
GCS-[7:0]	Input	<ul style="list-style-type: none"> • Chip select signals. These signals are always driven by a valid value during the bus cycle. • The CPU board makes the corresponding chip select signal active to specify the resources on the motherboard when the CPU board is the bus master. • Each chip select signal specifies the type of memory/I/O space and the width of the space (see Section 14.5).
GRD-	Input	<ul style="list-style-type: none"> • Read timing signal. This signal is asserted when the CPU board is the bus master. • This signal is not used by the motherboard. • If the CPU has an RD- command signal, that signal is usually connected.
GWR-	Input	<ul style="list-style-type: none"> • Write timing signal. This signal is asserted when the CPU board is the bus master. • This signal is not used by the motherboard. • If the CPU has a WR- command signal, that signal is usually connected.
GHOLD-	Output	<ul style="list-style-type: none"> • Bus hold request signal. • This signal is asserted low when the motherboard accesses the resources on the CPU board to acquire bus mastership. • If the GUSE_DIRECT_ACC- signal is high, the GHOLD- signal indicates to the CPU board that the motherboard has no resources that can be accessed. In this case, the CPU board does not have to support GHOLD-.
GHLDA-	Input	<ul style="list-style-type: none"> • Bus hold acknowledge signal. • This signal indicates that the CPU board releases bus mastership of GBUS to the motherboard. It is then asserted low. • The CPU board that asserts the GUSE_DIRECT_ACC- signal high can disconnect the GHLDA- signal.
GBREQ-	Input	<ul style="list-style-type: none"> • Bus mastership release request signal • When the motherboard has bus mastership from asserting GHLDA- low, the CPU board asserts GBREQ- low when it requires bus mastership. • If GBREQ- is asserted low and the motherboard is in bus cycle, GBLAST- must be asserted in the next micro cycle, the bus cycle must be completed in the next micro cycle, and GHOLD- must be deasserted. • GBREQ- is used to return bus mastership to the CPU board temporarily if the number of bursts in the bus cycle is large when the motherboard is the bus master, or if a bus cycle with a high priority such as a refresh cycle is pending on the CPU board.
GDMARQ-[3:0]	Output	<ul style="list-style-type: none"> • DMA request signals. Only two-cycle DMA is supported. Fly-by DMA is not supported. • These signals are asserted low if a DMA request is generated on the motherboard. • The CPU board must support all four DMA signals. The number of DMA signals that can be asserted at the same time and can be supported by the GDMAAK- signal depends on the CPU board. • The CPU board uses the DMAAK signal in preference to DMAAK-[3:2] if correspondence between all four GDMARQ- signals and GDMAAK- signals cannot be established.

Signal name	Input/output	Function
GDMAAK-[3:0]	Input	<ul style="list-style-type: none"> • DMA acknowledge signals. • These signals are asserted low to acknowledge DMA requests from the motherboard. • The CPU board uses the DMAAK signal in preference to DMAAK-[3:2] if correspondence between all four GDMARQ- signals and GDMAAK- signals cannot be established. • The motherboard is designed to operate even though there is no GDMAAK- signal.
GINTO-[3:0]	Output	<ul style="list-style-type: none"> • Interrupt request signals. • GINTO0- can be used as a level-sensitive signal. • Whether GINTO-[3:1] can be used as level-sensitive signals or edge-sensitive signals depends on the CPU board (since they may be directly connected to the CPU). The motherboard can support both level- and edge-sensitive signals. • Occurrence of an interrupt is indicated when these signals are low or on the falling edges of these signals.
GINTI-[1:0]	Input	<ul style="list-style-type: none"> • Interrupt request signals. • These interrupt signals are used to combine an interrupt on the CPU board with an interrupt on the other motherboard and return the combined signal to GINTO-[3:0]. • Usually, OUT0 and OUT1 of TIC (μPD71054) on the CPU board are connected. The motherboard can select the type of sensitivity and polarity of these interrupt signals.
GETC[7:0]	--	<ul style="list-style-type: none"> • CPU board dependent signals. • The contents of GETC[7:0], including the direction and contents of the signals, are determined by the CPU board. The CPU board uses these signals to exchange special signals with the motherboard.
GAHI_EN-	Input	<ul style="list-style-type: none"> • Upper address valid signal. • If this signal is low and if the CPU board is the bus master, the CPU board drives a valid value on GADDR[31:26]. If this signal is high, the CPU board does not drive a valid signal on GADDR[31:26], and the circuits on the motherboard perform processing with all of GADDR[31:26] low.
GMOTHER_DETECT-	Output	<ul style="list-style-type: none"> • Motherboard detection signal. • This signal is pulled up on the CPU board, and is connected to GND on the motherboard. The CPU board uses this signal when it must determine if the motherboard is connected (for example, time-over ready generation circuit of the CPU board).
GUSE_DIRECT_ACC-	Input	<ul style="list-style-type: none"> • If this signal is low, the CPU board has resources that can be accessed by the motherboard.
GCLK_LOW-	Input	<ul style="list-style-type: none"> • If this signal is low, the frequency of GCLK is 16.67 MHz or less. If it is high, the frequency of GCLK is 16.67 to 33.33 MHz. • The circuits on the motherboard use this signal to determine the number of wait cycles required for accessing the resources on the motherboard.
GBLOCK-[1:0]	Input	<ul style="list-style-type: none"> • Bus lock signals. These signals must be valid during a bus cycle and for bus cycles that must be locked. • If a bus lock signal is output by the CPU, the bus lock signal is connected to the motherboard using these pins. • The GBLOCK0- signal is valid for the GCS0- space. GBLOCK1- is valid for the GCS5- and GCS7- spaces.
+5V	Output	<ul style="list-style-type: none"> • Power supply. Supplies +5 V \pm5% from the motherboard to the CPU board.

Signal name	Input/output	Function
+12V	Output	<ul style="list-style-type: none">• Power supply. Supplies +12 V \pm10% from the motherboard to the CPU board. However, if the CPU board does not require +12 V, the motherboard does not have to supply +12 V.

14.3. PIN ASSIGNMENTS

The following table shows the GBUS pin assignments. Reserve indicates a reserved pin. N/C indicates that a pin is not connected.

No.	Signal name	No.	Signal name	No.	Signal name	No.	Signal name
1	+12V	2	+12V	3	GND	4	+5V
5	GADDR2	6	GADDR3	7	GADDR4	8	GADDR5
9	GADDR6	10	GADDR7	11	GND	12	+5V
13	GADDR8	14	GADDR9	15	GADDR10	16	GADDR11
17	GADDR12	18	GADDR13	19	GADDR14	20	GADDR15
21	GND	22	+5V	23	GADDR16	24	GADDR17
25	GADDR18	26	GADDR19	27	GADDR20	28	GADDR21
29	GADDR22	30	GADDR23	31	GND	32	+5V
33	GADDR24	34	GADDR25	35	GADDR26	36	GADDR27
37	GADDR28	38	GADDR29	39	GADDR30	40	GADDR31
41	GND	42	+5V	43	GBEN3-	44	GBEN2-
45	GBEN1-	46	GBEN0-	47	GND	48	+5V
49	GDATA31	50	GDATA30	51	GDATA29	52	GDATA28
53	GDATA27	54	GDATA26	55	GDATA25	56	GDATA24
57	GND	58	+5V	59	GDATA23	60	GDATA22
61	GDATA21	62	GDATA20	63	GDATA19	64	GDATA18
65	GDATA17	66	GDATA16	67	GND	68	+5V
69	GDATA15	70	GDATA14	71	GDATA13	72	GDATA12
73	GDATA11	74	GDATA10	75	GDATA9	76	GDATA8
77	GND	78	+5V	79	GDATA7	80	GDATA6
81	GDATA5	82	GDATA4	83	GDATA3	84	GDATA2
85	GDATA1	86	GDATA0	87	GND	88	+5V
89	GND	90	GW/R-	91	GBTERM-	92	GREADY-
93	GRESETI-	94	GADS-	95	GBLAST-	96	GWAITI-
97	GND	98	GCLK	99	GND	100	+5V
101	GCS0-	102	GCS1-	103	GCS2-	104	GCS3-
105	GCS4-	106	GCS5-	107	GCS6-	108	GCS7-
109	Reserve	110	Reserve	111	Reserve	112	Reserve
113	GRD-	114	GWR-	115	GND	116	+5V
117	GHOLD-	118	GHLDA-	119	GBREQ-	120	N/C
121	GDMARQ0-	122	GDMARQ1-	123	GDMARQ2-	124	GDMARQ3-
125	GDMAAK0-	126	GDMAAK1-	127	GDMAAK2-	128	GDMAAK3-
129	Reserve	130	Reserve	131	Reserve	132	Reserve
133	GND	134	+5V	135	GINTO0-	136	GINTO1-
137	GINTO2-	138	GINTO3-	139	GINTI0-	140	GINTI1-
141	GETC0	142	GETC1	143	GETC2	144	GETC3
145	GETC4	146	GETC5	147	GETC6	148	GETC7
149	Reserve	150	Reserve	151	GAHI_EN-	152	GMOTHER_DETECT-
153	GND	154	+5V	155	GUSE_DIRECT_ACC-	156	GCLK_LOW-
157	GRESETO-	158	GBLOCK0-	159	GBLOCK1-	160	N/C
161	N/C	162	N/C	163	N/C	164	N/C
165	N/C	166	N/C	167	N/C	168	N/C
169	N/C	170	N/C	171	N/C	172	N/C
173	N/C	174	N/C	175	N/C	176	N/C
177	GND	178	+5V	179	+12V	180	+12V

The following connectors are used:

- CPU board side connector → Kell 8817-180-170L
- Motherboard side connector (straight) → Kell 8807-180-170S
- Motherboard side connector (L angle) → Kell 8807-180-170L

14.4. PROCESSING OF UNUSED PINS

Signals that are not input to the GBUS motherboard are pulled up or down on the motherboard and can be unconnected on the CPU board. Signals that can be unconnected and the processing performed on the motherboard for those pins are shown below.

Signal name	Processing
GADDR[31:26]	<ul style="list-style-type: none"> If GADDR[31:26] are not used, GADDR[31:26] can be unconnected by making the GAHI_EN signal high or by disconnecting it. In this case, if the CPU is the bus master, all the bits of GADDR[31:26] are treated as 0 on the motherboard.
GWAITI-	<ul style="list-style-type: none"> Pull-up processing is performed.
GBLAST-	<ul style="list-style-type: none"> Pull-up processing is performed.
GBTERM-	<ul style="list-style-type: none"> Pull-up processing is performed.
GCS-[7:0]	<ul style="list-style-type: none"> Pull-up processing is performed.
GHLDA -	<ul style="list-style-type: none"> Pull-up processing is performed.
GBREQ-	<ul style="list-style-type: none"> Pull-up processing is performed.
GDMAAK-[3:0]	<ul style="list-style-type: none"> Pull-up processing is performed.
GINTI-[1:0]	<ul style="list-style-type: none"> Pull-up processing is performed.
GAHI_EN-	<ul style="list-style-type: none"> Pull-up processing is performed.
GUSE_DIRECT_ACC-	<ul style="list-style-type: none"> Pull-up processing is performed.
GCLK_LOW-	<ul style="list-style-type: none"> Pull-up processing is performed.
GBLOCK-[1:0]	<ul style="list-style-type: none"> Pull-up processing is performed.

14.5. ALLOCATING GCS-[7:0]

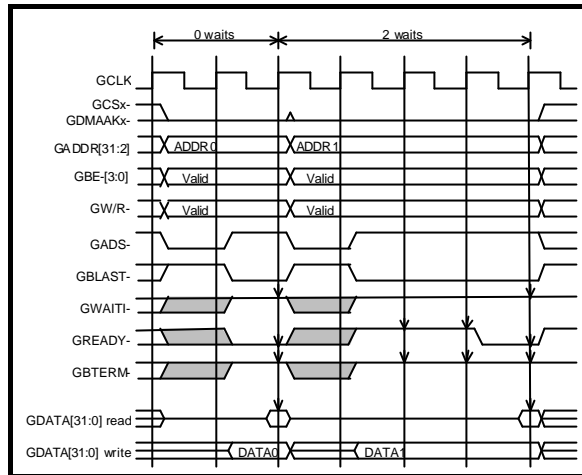
The following table shows the allocation of the chip select signals (GCS-[7:0]). All of the spaces can be accessed in a burst cycle. A space marked I/O under the heading "Recommended space" means that, if the CPU has an I/O space, it is recommended that the space be allocated as an I/O space. "Minimum range" indicates that the CPU board must allocate at least the indicated area for the corresponding chip select space. "Maximum range" indicates that, if the CPU board has an extra address range, addresses can be allocated for the indicated range.

Signal name	Recommended space	Minimum range	Maximum range	Remark
GCS0-	Memory	1M byte		Bus lock possible with GLOCK0-
GCS1-	Memory	2M bytes		Because a flash ROM is allocated to this space on the motherboard, the program must be able to be booted from this space, instead of from UV -EPROM on the CPU board, via a switch.
GCS2-	I/O	64K bytes		
GCS3-	Memory	64Kbytes	16M bytes	
GCS4-	I/O	64K bytes	16M bytes	
GCS5-	Memory	1M byte	2G bytes	Bus lock possible with GLOCK1-
GCS6-	I/O	512 bytes		
GCS7-	I/O	64K bytes	2G bytes	Bus lock possible with GLOCK1-

14.6. BUS CYCLE

14.6.1. Single Cycle

The following chart shows the single cycle when GBWAITI- and GBTERM- are always inactive and the CPU board is the bus master. If the motherboard is the bus master, the GCSx-, GDMAAK-, and GWAITI- signals are not used.

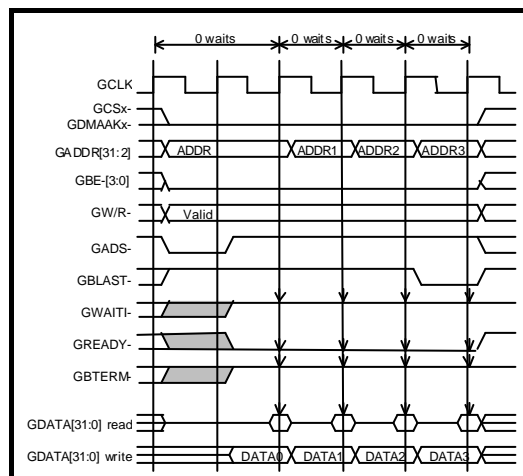


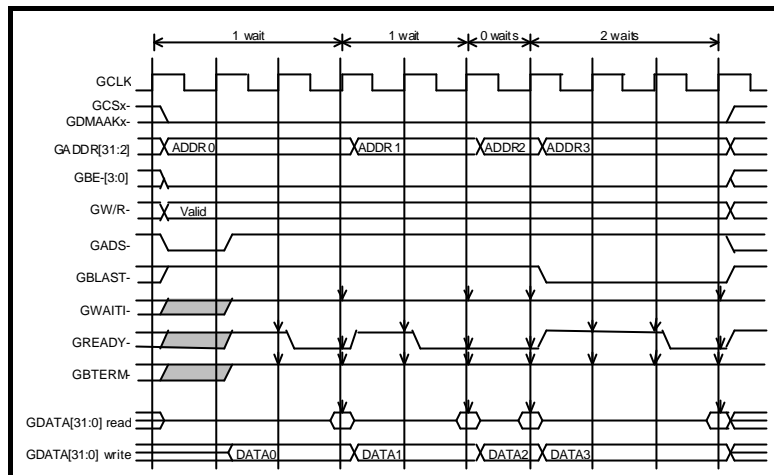
14.6.2. Burst Cycle

The following rules apply to a burst cycle:

- The addresses in the burst cycle can be in any sequence allowed by the GBUS specifications. However, the address sequence may be specified according to what is to be accessed.
- In a burst cycle, all of GBE-[3:0] must be active.
- The number of bursts (the number of micro cycles) is not limited. If the target of the access limits the number of bursts, use the GBTERM- signal (see Section 14.6.4) to request canceling of the burst.

The following chart shows the burst cycle when GBWAITI- and GBTERM- are always inactive and the CPU board is the bus master. If the motherboard is the bus master, the GCSx-, GDMAAK-, and GWAITI- signals are not used.





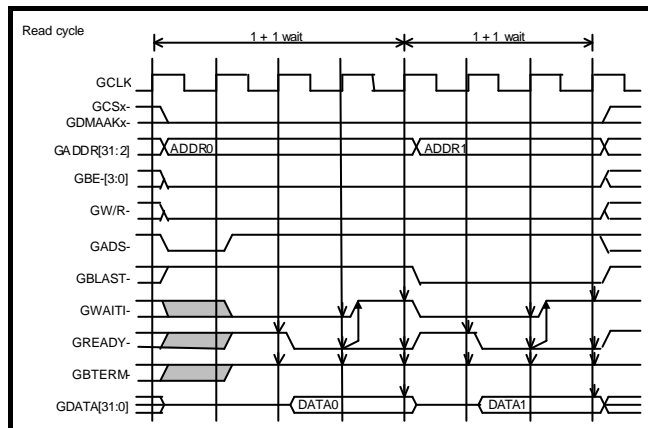
14.6.3. GWAITI-

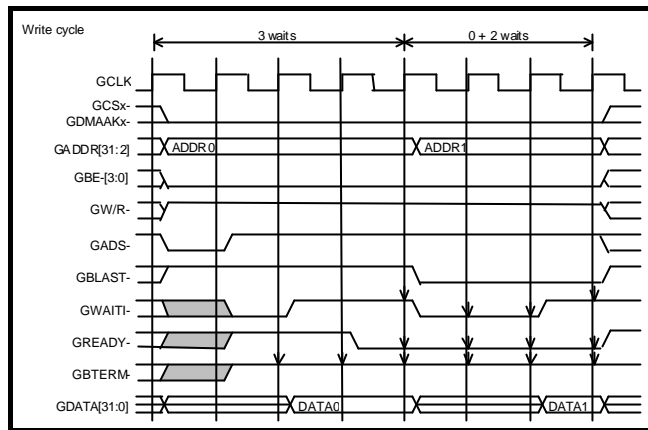
The GWAITI- signal can be used as follows in a cycle in which the CPU board is the bus master:

- To delay sampling of data by a specific number of clocks because the data cannot be sampled in the read cycle.
- To hold the target of an access by the specific number of clocks because data for the next micro cycle is not ready immediately after completion of the first micro cycle in the burst cycle of a write cycle.

In other words, the roles of the read cycle and write cycle are switched, but GREADY- and GWAITI- serve as data transmission ready and data reception ready signals.

The following chart shows that a wait cycle is inserted by the GWAITI- signal.



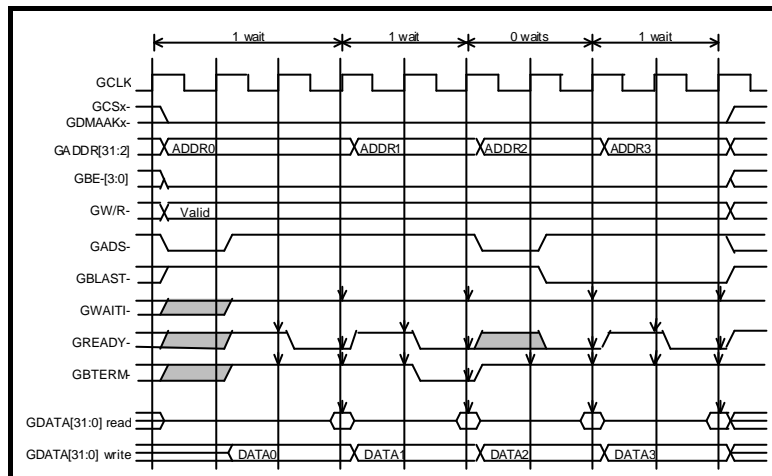


14.6.4. GBTERM-

If both the GBTERM- signal and GREADY- signal become active at the same time, the bus master completes the bus cycle after the current micro cycle ends, and then starts the burst cycle again by asserting GADS- active.

The GBTERM- signal is asserted active if the target of the access does not support burst cycles or accesses are made more than the supported number of bursts. Asserting the GBTERM- signal only without also asserting the GREADY- signal is not allowed.

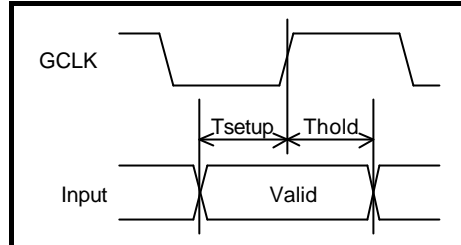
The following chart shows that the burst cycle is canceled by the GBTERM- signal.



14.7. TIMING

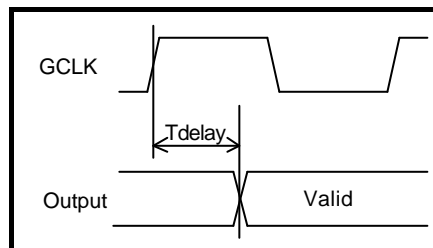
This chapter describes the timing of Midas lab's motherboard. The CPU board is designed to satisfy this timing.

14.7.1. Setup Time



Signal name	Tsetup Min. (ns)	Thold Min. (ns)
GADDR[31:2]	12	0
GBEN-[3:0]	8	0
GDATA[31:0]	7	0
GADS-	14	0
GREADY -	9	1
GWAITI-	14	0
GBLAST-	8	0
GBTERM-	8	1
GW/R-	10	0
GCS-[7:0]	14	0
GBREQ-	15	0
GDMAAK-[3:0]	6	0
GLOCK-[1:0]	12	0

14.7.2. Delay Time



Signal name	Tdelay Max. (ns)
GADDR[31:2]	21
GBEN-[3:0]	17
GDATA[31:0]	21
GADS-	15
GREADY -	15
GBLAST-	17
GBTERM-	16
GW/R-	15

- Memo -

RTE-NB85E-CB User's Manual

M883MNL01

Midas lab