NEC

Preliminary User's Manual

QB-703427

IECUBE for V850E/Dx3

Hardware

Target Devices:
V850E/DG3: µPD70(F)341x (x = 6/7)
V850E/DJ3: µPD70F342x (x = 1/2/3/4/5)
V850E/DL3: µPD70F3427

Document No. U18350EE1V2UM00
Date published 27/05/08
© NEC Electronics 2008
Printed in Germany

The information contained in this document is being issued in advance of the production cycle for the product. The parameters for the product may change before final production or NEC Electronics Corporation, at its own discretion, may withdraw the product prior to its production.
Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.

Legal Notes

- **The information contained in this document is being issued in advance of the production cycle for the product. The parameters for the product may change before final production or NEC Electronics Corporation, at its own discretion, may withdraw the product prior to its production.**

- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.

- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.

- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.

- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.

- NEC Electronics products are classified into the following three quality grades: "Standard", "Special", and "Specific". The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics products before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)
(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives anddistributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and otherlegal issues may also vary from country to country.

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668, Japan
Tel: 044 4355111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554,
U.S.A.
Tel: 408 5886000
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211 65030
http://www.eu.necel.com/

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908 691133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01 30675800

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 6387200

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02 667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven,
The Netherlands
Tel: 040 2654010

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27
ZhiChunLu Haidian District,
Beijing 100083, P.R.China
Tel: 010 82351155
http://www.cn.necel.com/

NEC Electronics Shanghai Ltd.
Room 2511-2512, Bank of China
Tower,
200 Yincheng Road Central,
Pudong New Area,
Shanghai 200120, P.R. China
Tel: 021 58885400
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
12/F., Cityplaza 4,
12 Taikoo Wan Road, Hong Kong
Tel: 2886 9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R.O.C.
Tel: 02 27192377

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253 8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku, Seoul,
135-080, Korea Tel: 02-558-3737
http://www.kr.necel.com/

Preface

Target Readers This manual is intended for users who design and develop application systems using the IECUBE for V850E/Dx3.

Purpose The purpose of this manual is to describe the proper operation of the QB-703427 and its basic specifications.

Organization This manual is broadly divided into the following parts.

Overview

Setup procedure

Cautions

How to read this manual It is assumed that the reader of this manual has general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers. This manual explains the basic setup procedure, so read this document before using the QB-703427 .

To learn about the basic specifications and operation methods:

Read this manual in the order of the Table of Contents.

To learn about software settings such as operation methods and command functions.

Read the user's manual of the debugger that is used:

Legend Symbols and notation are used as follows:

Weight in data notation : Left is high-order column, right is low order column

Active low notation : $\overline{xxx}$ (pin or signal name is over-scored) or

/xxx (slash before signal name)

Memory map address : High order at high stage and low order at low stage

Note Explanation of (Note) in the text

Caution Item deserving extra attention

Numeric notation :

Binary... XXXX or XXXB

Decimal... XXXX

Hexadecimal... XXXXH or 0x XXXX

Prefixes representing powers of 2 (address space, memory capacity)

K (kilo): 210 = 1024

M (mega): 220 = 10242 = 1,048,576

G (giga): 230 = 10243 = 1,073,741,824

Further information For further information see http://www.eu.necel.com.

Table of Contents

Chapter 1 Introduction

Introduction

Terminology The meanings of terms used in this manual are listed below.

Table 1-1 Terminology

Term	Meaning
Target device	Refers to the device targeted for emulation.
Target system	Refers to the system targeted for debugging.This includes the target program and the hardware created by the user. In a narrow sense, it means hardware only.

Related Documents When using this manual, refer to the following manuals.

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Table 1-2 Documents related to development tools (user's manuals)

Document Name Document		Number
QB-703427 In-Circuit Emulator		This manual
QB-703427 In-Circuit-Emulator Option Board Operating Precautions Target	Customer Notification	EEDT-OP-0030 - current version
CPDW9X/NT-CDR-V85X	GHS integrated development environment (PC, MS Windows based)	Ver.4.07 or - current version
V850 Series CPDW9X/NT-CDR-V85X Operating Precautions GHS MULTI 2000 Integrated DevelopmentEnvironment Version 3.X, 4.X	Customer Notification	U18068EE7V0IF00 or - current version
850eserv / Detailed information listing	Read Me File for 850ESERV	850eserv.txt - current version
IEQBUTL / IECUBE Utility	Read Me File for IECUBE Utility IEQBUTL	IEQBUTL_E2.11e .t xt or - current version

Note Download the documents from the NEC Electronics (Europe) GmbH web site.
URL: http://www.eu.necel.com/products/micro/

General Cautions on handling this product

1. NEC Electronics' warranty does not cover the following cases:

- When the QB-703427 is disassembled, reconstructed, or modified by the user
- When the QB-703427 receives a serious shock, e.g., by being dropped
- When the QB-703427 is used with excessive voltage or is stored outside the guaranteed temperature range or guaranteed humidity range
- When power is applied without the AC adapter, USB interface cable, or target system properly connected
- When the AC adapter cable, USB interface cable, or extension probe is unduly twisted or stretched
- When an AC adapter other than the one supplied with the QB-703427 is used
- When water is spilled on the QB-703427

2. Cautions on safe use

- The QB-703427 heats up (to approx. 50 to 60°C) when it operates for a long time. Take care not to burns yourself.
- Be very careful to avoid electric shocks. There is risk of electric shock if the product is used as described under 1.

Chapter 2 Overview

The QB-703427 (IECUBE) in-circuit emulator is used to emulate the V850E/Dx3 devices. The IECUBE allows hardware and software to be debugged efficiently in system development.

This manual describes the basic setup procedure, hardware specifications, system specifications and switch settings.

In this document the QB-703427 is also referred as IECUBE.

Figure 2-1 IECUBE Housing

2.1 Hardware Specifications

Table 2-1 Hardware Specifications

Item	Specification
Target device	V850E/Dx3
Target system interface voltage (unit: V)	MVDD5 = 3.0 V ~ 5.5 V, DVDD5 = 3.0 V ~5.5 V, BVDD5 = 3.0 V ~ 5.5 V, AVDD = 3.2 V ~ 5.5 V, SMVDD5 = 3.2 V ~ 5.5 V, VDD5 = 3.2 V ~ 5.5 V, VSS5 = BVSS5 = DVSS5 = SMVSS5 = MVSS5 =AVSS = 0 V
Maximum operating frequency	64 MHz + SSCG modulation range
Operating temperature range	10 to 40°C (without condensation)
Storage temperature range	-15 to 60°C (without condensation)
Package dimensions	See below
Power consumption: AC adapter for IECUBE	AC adapter for IECUBE 15 V, 1 A
Power consumption: Target system power supply	Lower than that of target device
Weight	475 g - 537 g
Host interface	USB interface (1.1 and 2.0)

2.2 IECUBE Dimensions

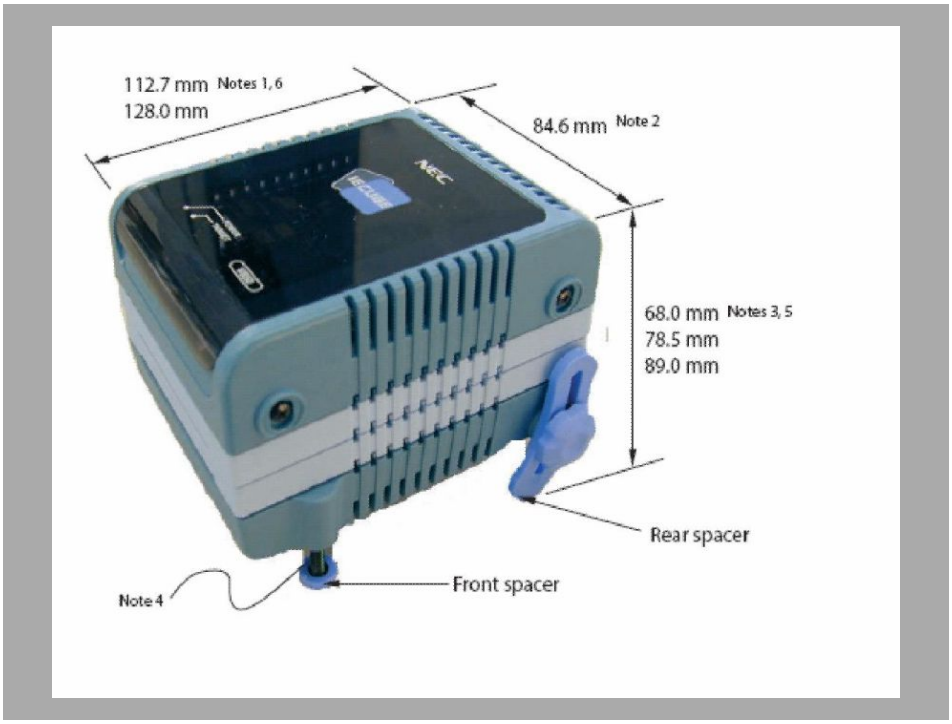


Figure 2-2 IECUBE Dimensions

Note 1. Excluding power supply switch.

2. Including screw for fixing rear spacer.

3. Shortest length of rear spacer (+30 mm max.).

4. Front spacer varies between 20 mm (max.) and 5 mm (min.).

5. Height (68.0 mm / 78.5 mm / 89.0 mm) depends on the number of boards used internally. This relates to 0,1 or 2 light blue plastic spacer used.

6. Rear fan increases length to 128.0 mm.

2.3 System Specifications

Table 2-2 System Specifications

Function		Specification
Emulation	Internal ROM	1 Mbyte (Maximum)
Memory capacity	Internal RAM	60 Kbyte (Maximum)
	External memory	16Mbyte (Maximum) on MM-board
Program execution function	Real-time execution function	Go, Start from here, Go & Go, Come here, Restart, Return out
	Non-real-time execution function	Step in, Next over, Slow motion
Break function	Hardware break	Execution: 10 points (Address) Access: 6 points (Data)
	Software break	2000 points (debugger related)
	Fail-safe break	Non-map, I/O illegal, write protect
	Other	Trace full break, Manual Break, Timer Over Flow Break
Trace function	Trace data type	Branch source PC, branch destination PC, all PCs, all execution data, access data, access address, R/W status, time stamp, DMA point (start/end)
	Trace mode	Speed Priority, Trace Priority
	Trace event	Delay trigger, section, qualify
	Memory capacity	256k frames
Real-time RAM monitor function		256 bytes x 8 points
Time measurement function	Measurement clock	Measurement-dedicated clock or CPU clock
	Measurement target	Program execution start to end Start event to end event
	Maximum measurement time	About 195 hours (when measurement-dedicated clock is used)
	Minimum resolution	20 ns
	Number of timers used for measurement	8
	Measurement result	Execution time (execution start to end) Max., min., average, pass count (between events)
	Other	Timer overflow break function (1 point)
Coverage function		Optional (under development)
Other functions		Mapping function, event function, register manipulation function, memory manipulation function

Caution Some of the functions may not be supported, depending on the debugger used.

2.4 Typical System Configuration

The system configuration when connecting the QB-703427 to a PC (or PC compatible) is shown below. Communication is based on USB Ver1.1 or Ver2.0 interface.

Connectors vary depending on the target device to be emulated.

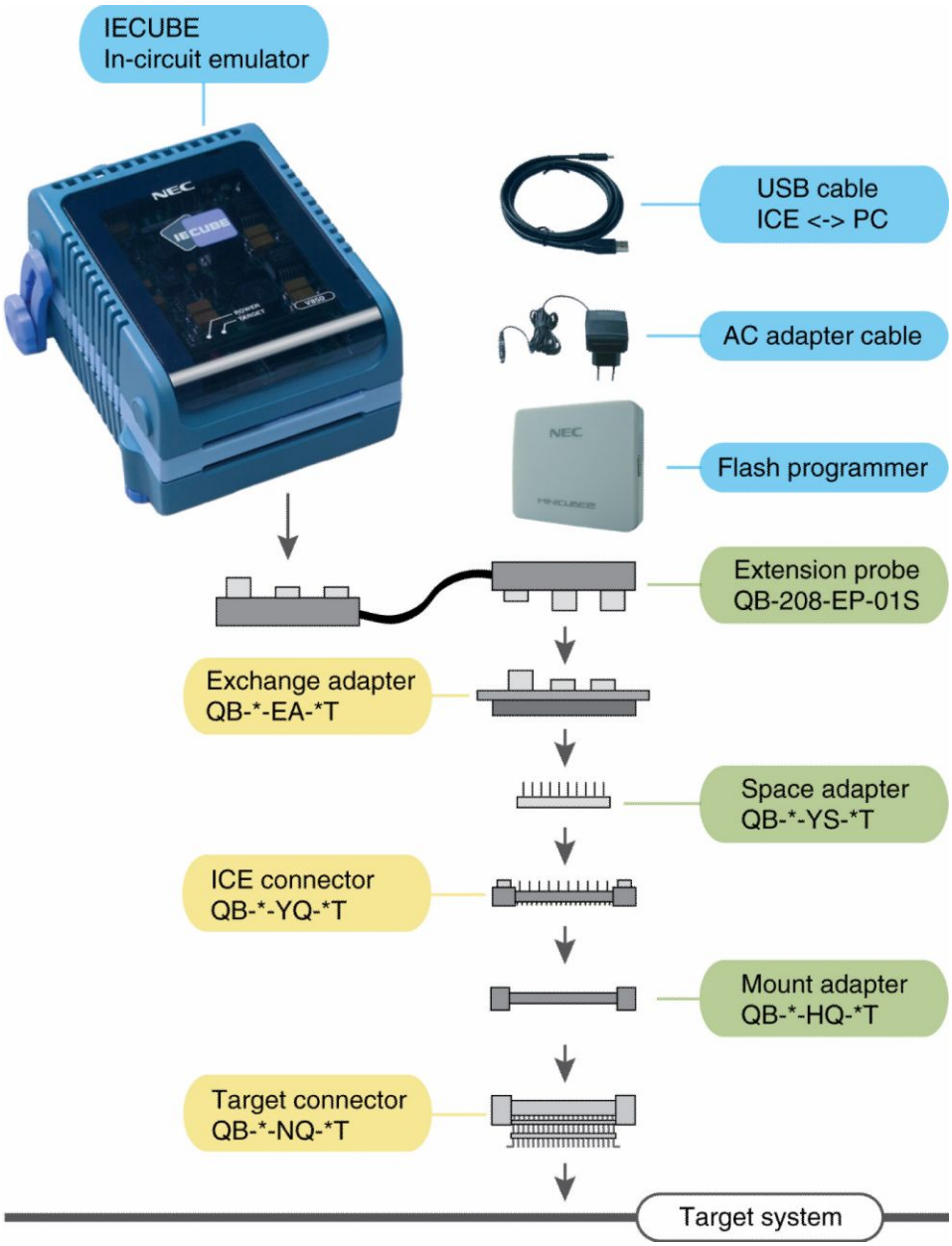


Figure 2-3 Typical System Configuration

See next chapter for the detailed information of the configuration and it's order codes for each device.

2.5 System Configuration

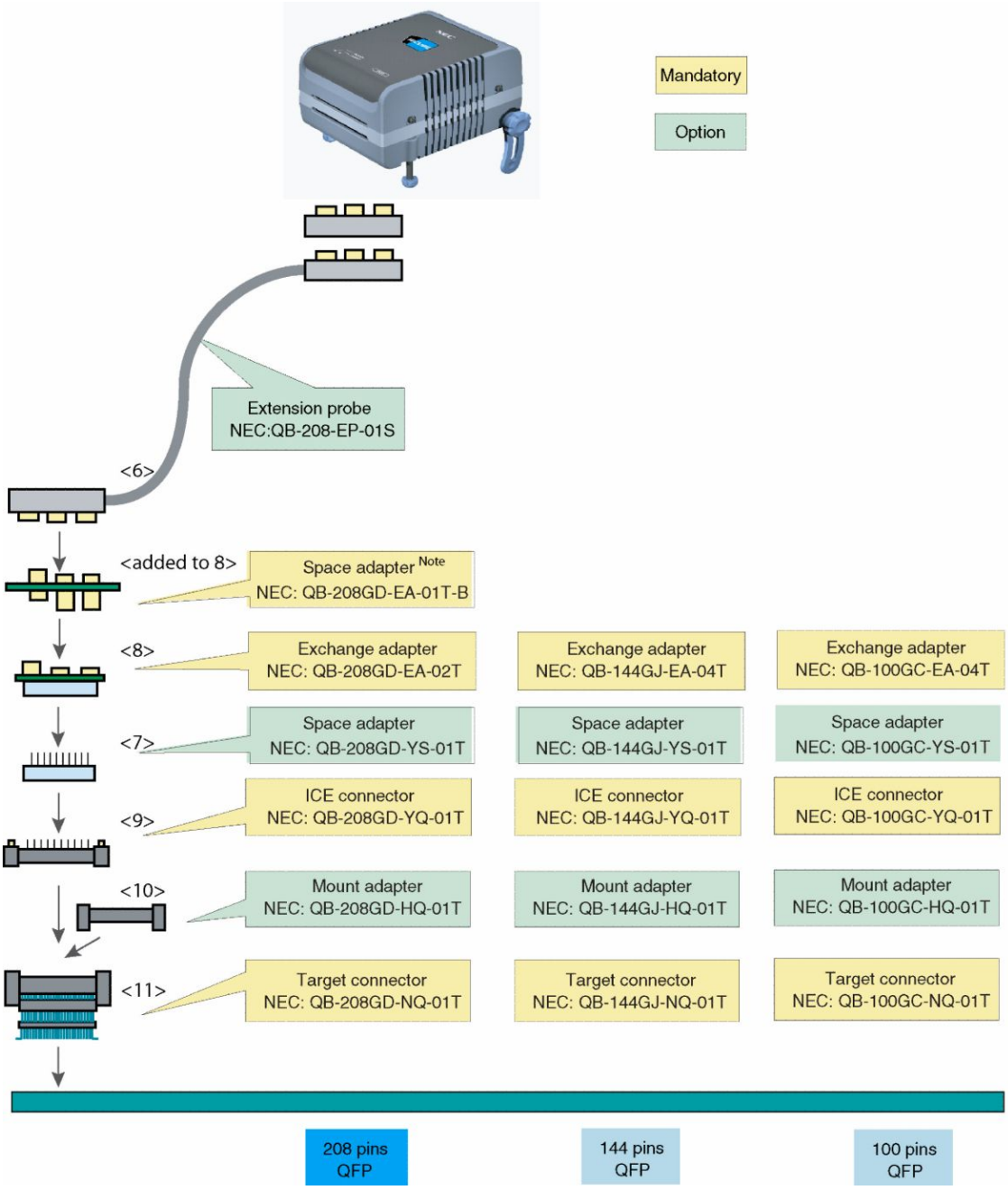


Figure 2-4 Structure of target adapter

Note Refer to the table below when selecting the exchange adapter. The above picture shows only one example.

Note When the 208pin exchange adapter is used without extension probe, a special space adapter is required. It is included in the package box of the QB-208GD-EA-02T. This is not required for the 144pin or 100pin exchange adapters.

Table 2-3 List of Probes and Connectors for Each Target Device

No.	Target Device	V850E/DG3	V850E/DJ3	V850E/DL3
	Package Type	uPD70(F)341xGC(A)-UEU-QS-AX (x == 6/7)	µPD70F342xGJ(A)-GAE-AX (x == 1/2/3/4/5)	µPD70F3427GD(A)-LML-AX
	Package dimension	100 LQFP/ 14mm x 14mm/0.5mm	144 LQFP/ 20mm x 20mm / 0.5mm	208 QFP/ 28mm x 28mm / 0.5mm
<6>	Extension probe	QB-208-EP-01S (coaxial type)		
<7>	Spacer adapter	QB-100GC-YS-01T	QB-144GJ-YS-01T	QB-208GD-YS-01T
<8>	Exchange adapter	QB-100GC-EA-04T	QB-144GJ-EA-04T	QB-208GD-EA-02T
<9>	YQ adapter	QB-100GC-YQ-01T	QB-144GJ-YQ-01T	QB-208GD-YQ-01T
<10>	Mounting adapter	QB-100GC-HQ-01T	QB-144GJ-HQ-01T	QB-208GD-HQ-01T
<11>	Target connector	QB-100GC-NQ-01T	QB-144GJ-NQ-01T	QB-208GD-NQ-01T

Note An order for the QB-703427 does not include the extension probe, the exchange adapters and target connectors. These parts listed under No. <6> to <11> will sold separate.

Note The QB-208GD-EA-02T (see table No. <8>) includes one space adapter QB-208GD-EA-01T-B required for mounting the QB-208GD-EA-02T directly to the IECUBE without a probe.

Caution Order the corresponding exchange adapter for the selected target device.

Adapter assembly : The QB-144GJ-YQ-01T and QB-208GD-EA-02T includes a non-standard YQGUIDE called YQGUIDE-S3. The YQGUIDE-S3 must be mounted between YQ and EA.

Different adapter pin size : Order the adapter for the related pin size of the selected device. Refer for the selection to the table above.

2.6 Packing Contents

Make sure that the following items are included in the QB-703427 package:

Hardware:

(1) QB-703427 in-circuit emulator

(2) AC adapter

(3) USB interface cable

(4) MINICUBE2 flash programmer (and asseccories)

Documentation:

(8) See QB-703427 package contents list

Chapter 3 IECUBE Setup Procedure

This chapter describes the procedure for setting up the QB-703427. Perform setup using the following procedure.

(1) Clock settings

A 4.000 MHz crystal is mounted at shipment for main clock.

A 32.768 kHz crystal is mounted at shipment for sub clock.

The internal low-speed on-chip oscillator clock is generated internally only.

There is no need to change the setting. When different type of crystal (e.g. same as on target system) is used, follow instructions given in this manual.

(2) Target device setting

It is assumed that the IECUBE target device is the V850E/Dx3 at shipment.

There is no need to change the setting when emulating the V850E/Dx3. If a different emulation option is used, follow instructions given in this manual.

(3) Software setup

(4) Mounting and connecting connectors

(5) Connecting IECUBE to target system

- When extension probe (QB-208-EP-01S) is used
- When extension probe (QB-208-EP-01S) is not used

(6) Connecting USB interface cable and AC adapter

(7) Power application/shutdown

3.1 Names and Functions of Hardware

Description of hardware interfaces and setup of the IECUBE:

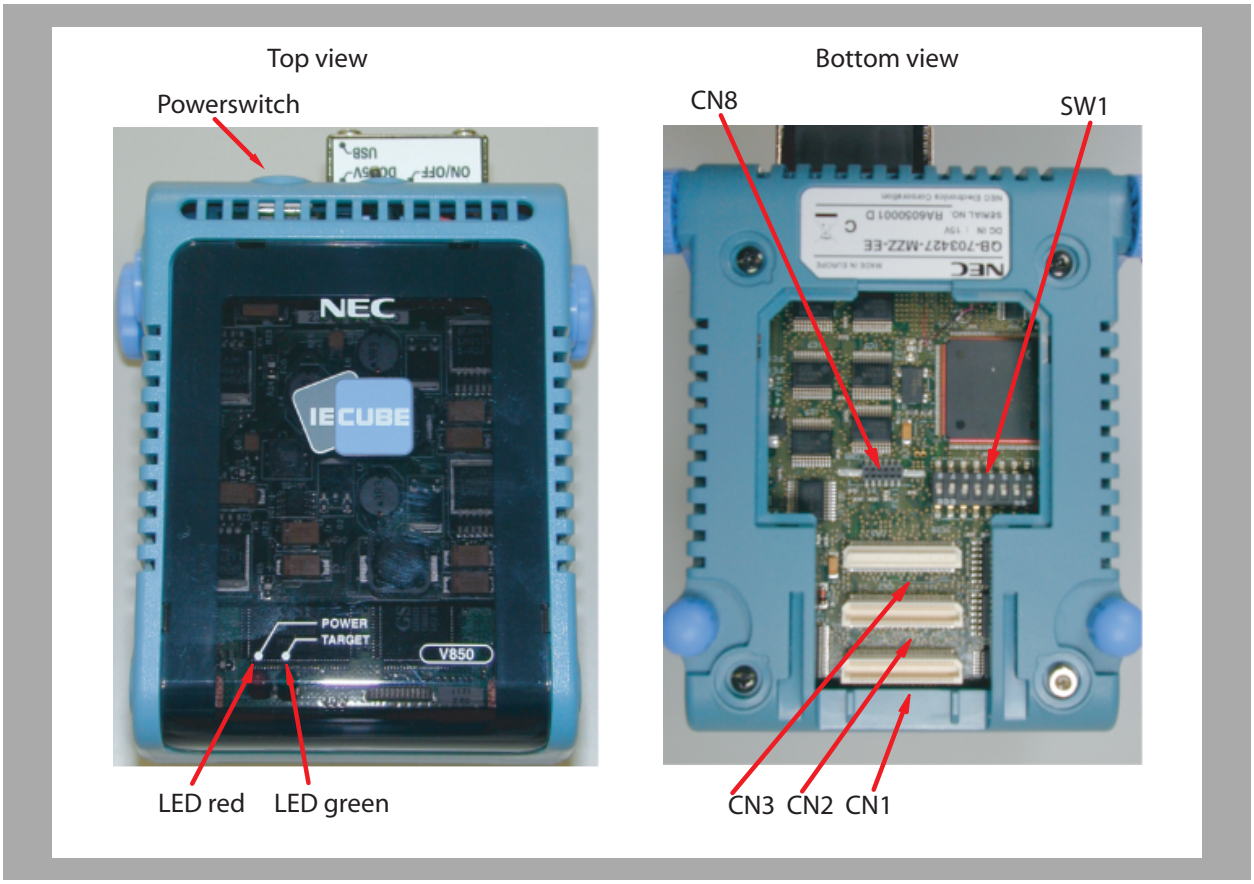


Figure 3-1 Names and Functions

(1) CN1, CN2, CN3

These connectors are used to connect the exchange adapter or extension probe.

(2) Clock adapter board connector (for clock) CN8

This clock adapter board is used for mounting the crystal.

4.000 MHz crystal and capacitors for configuring an oscillator circuit are mounted at shipment.

(3) SW1

This switch is set to select a target device or emulation function other than the default setting.

It is set to default at shipment.

(4) POWER (red LED)

This LED indicates whether power is applied to the IECUBE.

LED Status	IECUBE Status
Lit	The power supply is on.
Extinguished	The power supply is off, or the AC adapter is not connected to IECUBE.
Blinking	Internal error has occurred. (Contact an NEC Electronics sales representative or distributor.)

(5) TARGET (green LED)

This LED indicates whether power is applied to the target system.

LED Status	Target System Status
Lit	The power supply to the target system is on.
Extinguished	The power supply to the target system is off, or the target system is not connected.

(6) Power supply switch

This is a power switch for IECUBE. This switch is turned off at shipment.

3.2 Removing Acrylic Board

Remove the acrylic board underneath the IECUBE to change the settings of jumpers or clocks.

Remove the acrylic board by lifting it up.

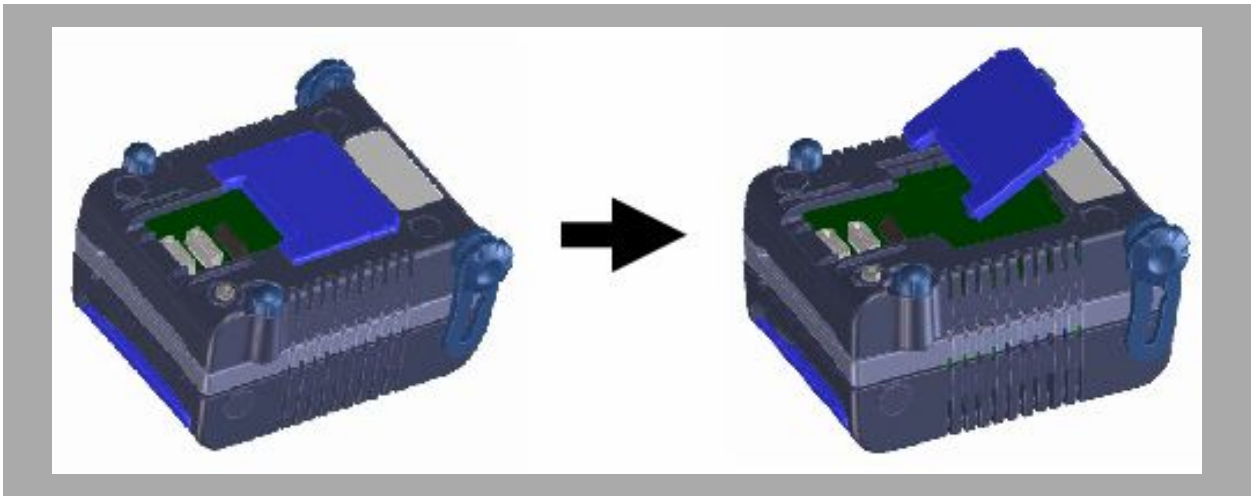


Figure 3-2 Removing Acrylic Board

3.3 Clock Settings

Note Default setting at shipment: clock adapter board is plugged in.There is no need to change the clock settings for normal operation. Change as described below only if special settings are required.

3.3.1 Overview of clock settings

Note Default setting at shipment: clock adapter board is plugged in.There is no need to change the clock settings for normal operation. Change as described below only if special settings are required.

Three methods are available for setting the clock.

(1) Use the 4.000 MHz/32.768 kHz crystals mounted on the clock board on the IECUBE as the internal clock.

(2) Mount different 4.000 MHz/32.768 kHz types of crystals on the clock board on the IECUBE. This option is only available if the same type of crystal is used as on the target system.

(3) Use clock input from target system without using clock board on IECUBE[Note]

Note The IECUBE does not support clock input from the target system. The function cannot be guaranteed if the target system clock is used.

3.3.2 How to set clock

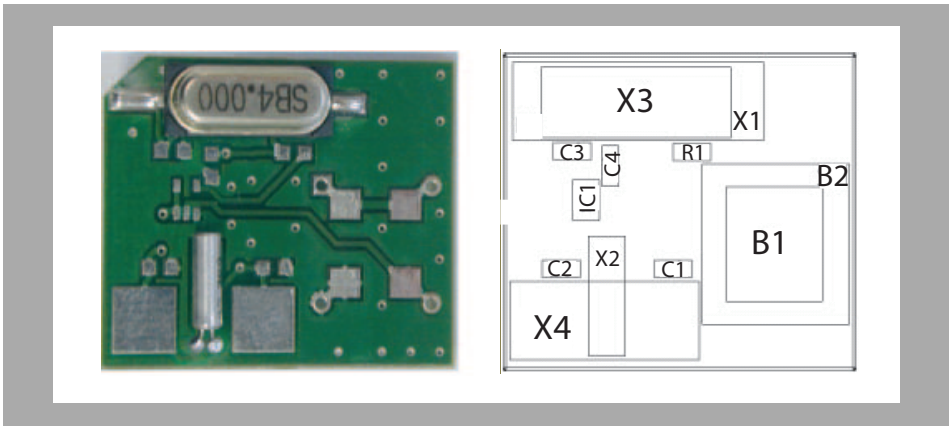
Two methods are available for setting the clock.

Clock to Be Used	Parts Board
Use the 4.000 MHz / 32.768 kHz crystal mounted on the clock board as internal clock.	Use the factory setting. The clock board is mounted.
Mount different crystals on the clock board and use as internal clock. The crystal frequency must be the same as the 4.000 MHz target device	The clock board crystal is of different type. The clock board is mounted after modification.

Settings other than above are prohibited.

3.3.3 How to change crystal

<1> Remove the clock adapter board, taking care not to damage IECUBE. Replace X1 and X2 by another type. Remount clock adapter board again.



<2> Default mounted

<3> X1: crystal 4.000 MHz SMD type *

X2: crystal 32.768 kHz DIL type *

* 10 pF capacitor on X1,X2 and XT1,XT2 are assembled on the I/O-board.

<4> Solder the crystal and capacitor on the parts board supplied with IECUBE as follows.

<5>

X1 replace by X3: crystal DIL type

X1 replace by B1: oscillator DIL type, additional IC1 and R1 must be assembled

X1 replace by B2: oscillator SMD type, additional IC1 and R1 must be assembled

X2 replace by X4: crystal SMD type

Reference(s)	Value
CN1	SAMTEC/TFM-110-02-S-D-AK
X1	4.000 MHz
X2	32.768 kHz
X3	4.000 MHz
X4	32.768 kHz
IC1	IDT74ALVC1G125DY
B2	4.000 MHz / 3.3 V
R1	R33
C1, C2, C3, C4	10 pF
B1	4.000 MHz / 3.3 V

3.4 Target IECUBE Settings

The SW1 setting varies depending on the target device / selected options. Settings other than below listed are prohibited.

Table 3-1 DIP-switch settings of QB-703427

Switch position OFF/ON				Function	Comment
SW1.1				Target power detection	
OFF				Check only target VDD	
ON				Check all target voltages	Default
SW1.2				ADC voltage selection	
OFF				Always use internal 5.0V	
ON				Use AVDD/AVREF from target if target available	Default
SW1.3	SW1.4	SW1.5	SW1.6	Technology / Package	
OFF	OFF	ON	ON	3420-3425 / 144 pins	Default
OFF	ON	ON	ON	reserved / 144 pins	
ON	ON	ON	ON	reserved / 208 pins	
OFF	OFF	OFF	ON	3426 / 144 pins	
ON	OFF	ON	OFF	3427 / 208 pins	
- all others -				Not supported	
SW1.7				Dithering at 64MHz	
OFF				Dithering at 64MHz is disabled/ masked	Default
ON				Dithering at 64MHz is allowed	
SW1.8				Unused	
OFF				-	
ON				-	Default

Caution Changing the default setting on SW1.7 may result in non error-free operation.

Note The Technology / Package setting for 3426 / 144 pins is reserved for a different emulator QB-703426.

The QB-703427 is delivered with the setting shown in the table below. This setting selects the mode for the emulation of the µPD70F3425 device. For emulation of other devices change the setting as described in the table above.

Table 3-2 Default DIP-switch settings (delivery) of QB-703427

SW1-1	SW1-2	SW1-3	SW1-4	SW1-5	SW1-6	SW1-7	SW1-8
On	On			On	On		On
		Off	Off			Off	

3.5 Software Setup

3.5.1 Debugger operation (Green Hills)

See the user's manual of the Multi - debugger (Green Hills) and the IECUBE Setup Manual.

Software versions

To use this tool, the following version of tools are required:

- Green Hills Multi V4.0.7 (or current)
- 850eserv (current)
- EXEC (current)

3.5.1.1 Connecting to target

To connect to the emulator with GHS Multi debugger, use the following command line:

(For complete information about option settings and configuration commands refer to the V850/850E ICE Server Reference Manual)

```
// configuration command: To connect to the target, use the command

connect 850eserv -iecube -tc


// option -iecube:  Connects through USB to IECUBE

// option -tc:   Specifies that the target board be connected to IECUBE.
```

If you specify this option, it detects unusual power status. Be sure to power on. (refer to GHS V850/850E ICE Server)

3.5.1.2 Specifying clock speed

To set the clock, use the following command line:

(For complete information about option settings and configuration commands

refer to the V850/850E ICE Server Reference Manual)

```
//configuration command: //Sets the target's minimum operating frequencies

target dclock 4000 32768 swoff


//DCLOCK [main_clock sub_clock]

//main_clock: specifies main clock per kHz

//sub_clock: specifies sub clock per Hz

//swon Uses Sub clock

//swoff Uses Main clock (default)
```

3.5.1.3 Mapping FCAN register space

To map the FCAN register space, use the following command line:

(For complete information about option settings and configuration commands refer to the V850/850E ICE Server Reference Manual)

```
// configuration command:  map aFCAN/PIO area to 0x3FEC000- 0x3FEFFFF

target piobase 0x3FEC000;

target map U2 = 0x3F00000, 0x3FFFFFF;


//MAP (U2=start,end)
```

The programmable peripheral I/O area needs to be mapped. If it is not mapped, the debugger will cause a break and issue a warning “Guard area access” when the CAN SFRs are accessed.

Map the memory as U2 (user memory, no target voltage checking).

Note that even though the PPIO area is only 12kB, a 1MB area must be mapped, as this is the smallest mappable area.

3.5.1.4 Mapping MM-board memory

To map the external memory of the MM-board (16Mbyte maximum), use the following command line:

(For complete information about option settings and configuration commands refer to the V850/850E ICE Server Reference Manual)

```
// configuration command:  map MM-board area to 0x00200000, 0x002FFFFF

target map W = 0x00200000, 0x002FFFFF CS4;


//MAP (W=start,end)
```

The emulator read/write memory area needs to be mapped. If it is not mapped, the debugger will cause a break and issue a warning “Guard area access” when the memory is accessed.

Map the memory as W (specifies emulator read/write memory, no target voltage checking).

CS0-CS7 specifies the chip selection domain mapped at the time of W or R specification.

A 1MB area must be mapped, as this is the smallest mappable area.

3.5.1.5 Download speed

To increase the download speed:

The download speed can be increased (main clock) to a higher frequency. Use the “hspload” command to allow fast download speed.

```
// configuration command:  use high-speed clock instead of internal high-
// speed on-chip oscillator

HSPLOAD ON



//ON Allows high-speed download at HSPLOAD.

//OFF Does not allow high-speed download at HSPLOAD.(default)
```

3.5.1.6 MULTI Resource file (RC-file)

Project dependent section

This is an example of what the start-up script may look like.

Note that this script sources the universal section listed below

```
//
//  sample.rc
//
//  Sample start-up script for QB-703425
//  (C) Copyright by NEC Electronics (Europe) GmbH
//
//  Revisions:
//  2007/04/05   Initial
//
// ***************************************************************************


connect 850eserv -df=DF3425J.800 -iecube -tc;

target dclock 4000 32768 swoff;
target rst;


// include device specific functions
<QB-703425.rc;


// load program to memory
QB_LoadCode();


// ********************************************************************* EOF ***
```

Universal section

This section may be used for every project, as it contains only mandatory settings that are project independent.

```
//
//  QB-703425.rc
//
//  Macros for uPD70F3425  configuration
//  (C) Copyright by NEC Electronics (Europe) GmbH
//
//  Usage:
//  Just include (<) this file and use the macros and symbols in project
//  specific .rc-file. Do not edit this file.
//
//  Revisions:
//  2007/04/13    Initial
//
// ***************************************************************************
```

```
// *** QB_Init ***
// function: performs device-specific but standard initialisation
define QB_Init()  {

  // map aFCAN/PIO area to    0x3FEC000- 0x3FEFFFF
  target piobase            0x3FEC000;
  target map U2  =        0x3F00000, 0x3FFFFFF;


  // register setup for IECUBE memory controller
target m h c 0xfffff48e=0xaaaa; // LBS tool setup only, used for MM-board
target m h c 0xfffff04a=0x1f; // PMCCT
// PCT0 - WRZO == device, general use
// PCT1 - WRZ1 == tool setup only, used for MM-board
// PCT2 - WRZ2 == tool setup only, used for MM-board
// PCT3 - WRZ3 == tool setup only, used for MM-board
// PCT4 - RDZO == device, general use

  // map emulator read/write memory  area to 0x00200000, 0x002FFFFF
target map W = 0x00200000, 0x002FFFFF CS4;



  // enable high-speed clock for downloading code
  target hspload on;
 }
QB_Init();    // do it now



// *** QB_LoadCode ***
// function: downloads code to emulator (after enabling high-speed download)
define QB_LoadCode()  {

  target hspload on;    // use high-speed clock instead of
               // internal on-chip oscillator when downloading code
               // this speeds-up start-up a lot
  load;           // download the code to the emulator
  target rst;       // issue a reset, so the startup code is executed
               // (starting at 0x0000000)
}


// ********************************************************************* EOF ***
```

3.5.1.7 Messages at Debugger start

(1) Details:

When the debugger is started, the following warning or error may occur depending on the setting of the debugger and the status of the target system. This is because the status of the target system is not in accordance with the setting of the debugger. If a warning or error occurs, check the status of the target system or the setting of the debugger. It is recommended that the conversion adapter be connected to the IECUBE even when the target system is not connected. If the conversion adapter is not connected, the value of the input port may not be correctly read.

Table 3-3 GHS MULTI Messages

Error Message	""-tc" of 850eserv Start Option		Target System Connection		Exchange Adapter		Target System Power	
	W/-tc	W/O -tc	Connect	Not Connect	Use	Not used	ON	OFF
Check the target power on. Or please delete "-tc" option.	x							x
Check the exchange adapter is connected.		x		x		x		x
Remove the target. Or please add "-tc" option and power on the target.		x	x					x
Power off and remove the target. Or please add "-tc" option.		x					x	

3.5.2 Debugger operation (IAR)

See the user's manual of the IAR - debugger and the IECUBE Setup Manual.

Software versions

To use this tool, the following version of tools are required:

- IAR Embedded Workbench V3.40A (or current)
- EXEC (current - included in the Embedded Workbench package)

3.5.2.1 Messages at Debugger start

(1) Details:

When the debugger is started, the following warning or error may occur depending on the setting of the debugger and the status of the target system. This is because the status of the target system is not in accordance with the setting of the debugger. If a warning or error occurs, check the status of the target system or the setting of the debugger. It is recommended that the conversion adapter be connected to the IECUBE even when the target system is not connected. If the conversion adapter is not connected, the value of the input port may not be correctly read.

Table 3-4 IAR Messages

Error Message	Emulator Hardware setup window		Target System Connection		Exchange Adapter		Target System Power	
	Clock source External	Clock source Internal	Connect	Not Connect	Use	Not used	ON	OFF
The target power is turned off. Failed to set clock Session aborted! The emulator configuration needs to be corrected Press YES to enter Emulator Hardware Setup again Pressing NO will stop debugger	x		x		x			x
The target power is turned off Failed to set clock Session aborted! The emulator configuration needs to be corrected Press YES to enter Emulator Hardware Setup again Pressing NO will stop debugger	x			x	x			x
Please check the connection of the exchange adapter.	x			x		x		x

Error Message	Emulator Hardware setup window		Target System Connection		Exchange Adapter		Target System Power	
Forced break is disabled. Use Reset to stop the emulator Press OK to continue Could not shut down the debug session		x	x		x			x

3.5.3 Debugger operation (ID850)

See the user's manual of the ID850 - debugger and the IECUBE Setup Manual.

Software versions

To use this tool, the following version of tools are required:

- ID850QB Integrated Debugger V3.40 (or current)
- EXEC (current)

3.5.3.1 Messages at Debugger start

(1) Details:

When the debugger is started, the following warning or error may occur depending on the setting of the debugger and the status of the target system. This is because the status of the target system is not in accordance with the setting of the debugger. If a warning or error occurs, check the status of the target system or the setting of the debugger. It is recommended that the conversion adapter be connected to the IECUBE even when the target system is not connected. If the conversion adapter is not connected, the value of the input port may not be correctly read.

Table 3-5 ID850QQB Messages

Error No.	Error Message	"Target" Field of ID850QB Configuration Window		Target System Connection		Exchange Adapter		Target System Power	
		Connect	Not Connect	Connect	Not Connect	Use	Not used	ON	OFF
Ff606	Check connection with the target and turn on power to the target.	x							x
Wf607	Check the connection of the conversion adapter.		x		x		x		x
Ff608	Disconnect the target.		x	x					x
Ff609	Turn off power to the target and disconnect the target.		x					x	

3.6 Mounting and Connecting Connectors

3.6.1 Usage

(1) When mounting NQPACK144SD to target system

<1> Coat the tip of four projections (points) at the bottom of the NQPACK144SD with two-component type epoxy adhesive (cure time longer than 30 min.) and bond the NQPACK144SD to the target system. If not bonded properly, the pad of the printed circuit board may peel off when the emulator is removed from the target system. If the lead of the NQPACK144SD does not coincide with the pad of the target system easily, perform step <2> to adjust the position.

<2> To adjust the position, insert the guide pins for position-adjustment (NQGUIDE) provided with NQPACK144SD into the pin holes on upper side of NQPACK144SD. The diameter of a hole is φ = 1.0 mm. There are three blind holes.

<3> After setting the HQPACK144SD, solder NQPACK144SD to the target system. Follow this sequence to avoid that flux or solder sputter adheres to the contact pins of the NQPACK144SD.

• Recommended soldering condition…Reflow : 240°C, 20 sec. max.

Partial heating : 240°C, 10 sec. max. (per pin row)

<4> Remove the guide pins.

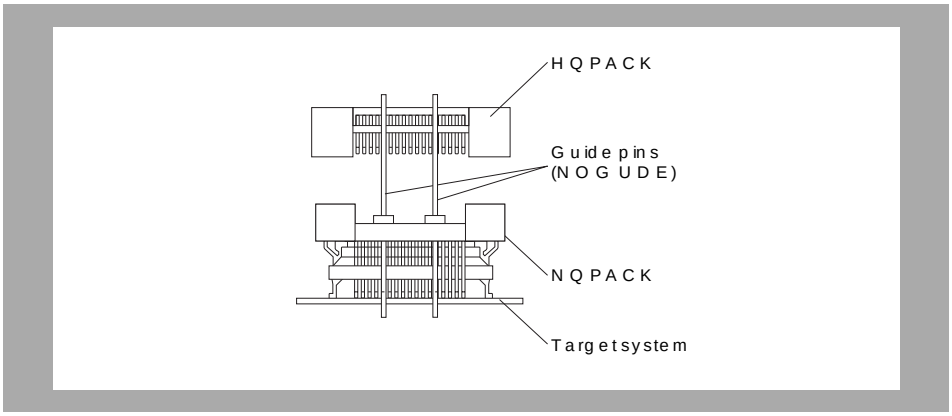


Figure 3-4 Mounting of NQPACK144SD

Note NQPACK144SD: Connector for target connection
HQPACK144SD: Cover for device installation

Mounting device

Caution Check for abnormal conditions such as resin burr or bent pins before mounting a device to the NQPACK144SD. Also check that the hold pins of the HQPACK144SD are not broken or bent before mounting the HQPACK144SD. Fix any broken or bent pins with a thin, flat tool such as a blade.

<1> Make sure that the NQPACK144SD is clean and the device pins are parallel (flat) before mouting a device to the NQPACK144SD. After mounting the NQPACK144SD to the target board, set the device and HQPACK144SD.

<2> Use the screws provided with the HQPACK144SD (four locations: M2 × 6 mm) to secure the HQPACK144SD, device, and NQPACK144SD. Tighten the screws in a crisscross pattern with the provided screwdriver or driver with torque gauge (avoid tightening strongly only one screw). Tighten the screws with 0.55 kg f cm (0.054 N m) max. torque. Excessive tightening may diminish conductivity. At this time, each pin is fixed inside the plastic wall dividers by the contact pin of the NQPACK144SD and the hold pin of the HQPACK144SD. These pins cannot cause a short with pins of neighboring devices.

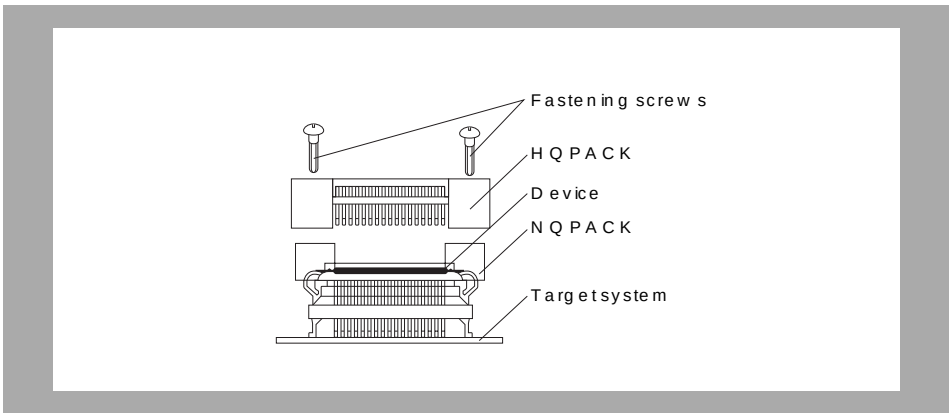


Figure 3-5 Mounting Device

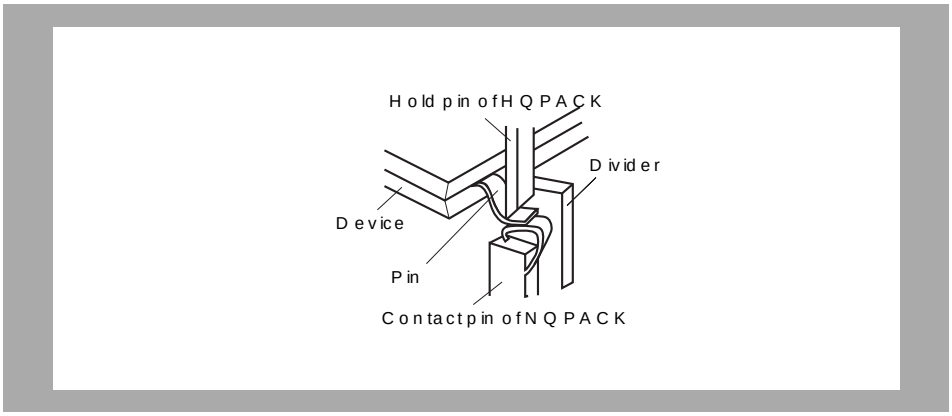


Figure 3-6 NQPACK144SD and Device Pin

3.6.2 Cautions on handling connectors

(1) When taking connectors out of the case, remove the sponge while holding the main unit.

(2) When soldering the NQPACK144SD to the target system, cover the HQPACK144SD to protect it against splashing flux.

• Recommended soldering conditions

Reflow : 240°C, 20 sec. max.

Partial heating : 240°C, 10 sec. max. (per pin row)

(3) Check for abnormal conditions such as resin burr or bent pins before setting a device to the NQPACK144SD. Also check that the hold pins of the HQPACK144SD are not broken or bent before mouting HQPACK144SD. Fix any broken or bent pins with a thin, flat plate such as a blade.

(4) When securing the YQPACK144SD (connector for emulator connection) or HQPACK144SD to the NQPACK144SD with screws, tighten the four screws temporarily with the provided screwdriver or driver with torque gauge, then tighten the screws in a crisscross pattern (with 0.054 N m max. torque).

Applying excessive torquw to one screw only may diminish conductivity.

If conductivity is diminished after tightening, stop tightening, remove the screws and check whether the NQPACK144SD is stained and make sure the device pins are parallel.

(5) Device pins do not have high strength. Repeatedly connecting to the NQPACK144SD may cause pins to bend. Check for and adjust any bent pins when mounting a device to the NQPACK144SD.

3.7 Connecting IECUBE to Target System

3.7.1 Connection without using extension probe (QB-208-EP-01S)

The IECUBE can be connected to the target system without using the extension probe. When connecting the IECUBE and the target system, adjust the height of IECUBE using the rear spacer so that no stress is applied to the exchange adapter and target connector. Also take care to maintain insulation with the target system.

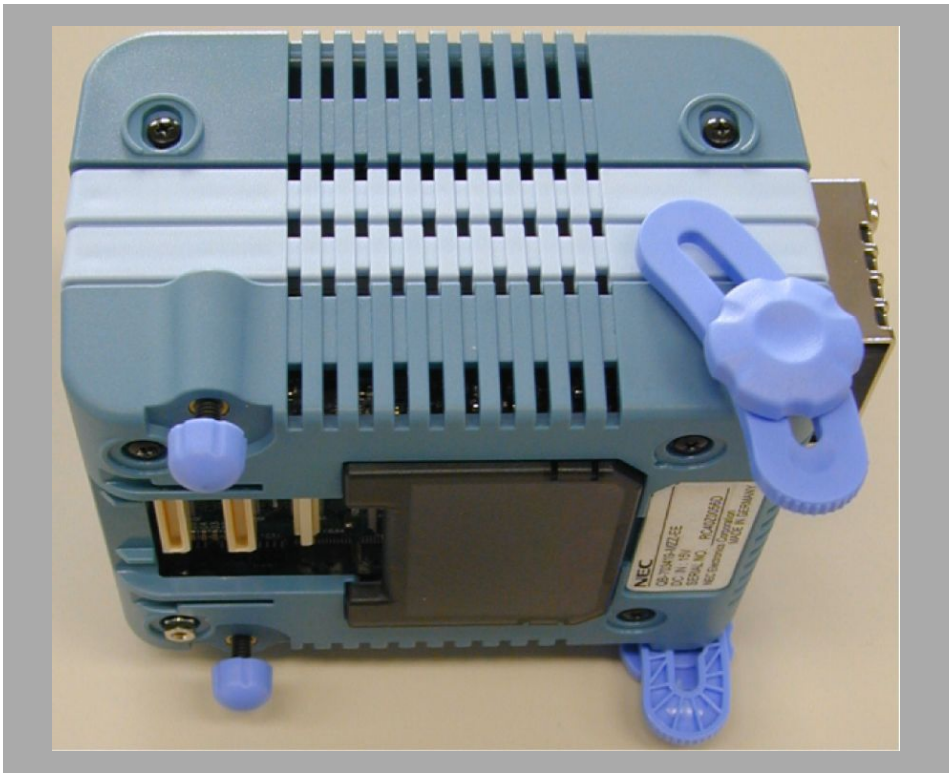


Figure 3-7 Connection without using Extension Probe

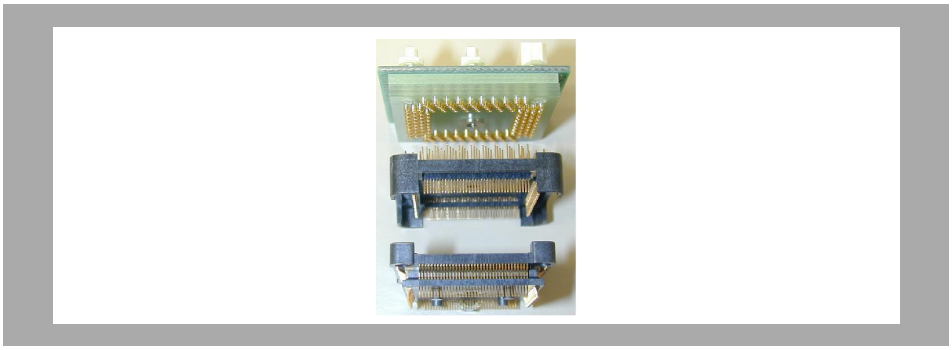


Figure 3-8 Connector Stack Up

3.7.2 Connection using extension probe (QB-208-EP-01S)

When using the extension probe (QB-208-EP-01S), connect the IECUBE and the target system as follows:

(1) Connecting probe holder

Use the probe holder (included with IECUBE) for connecting the extension probe to IECUBE. Connect as shown below.

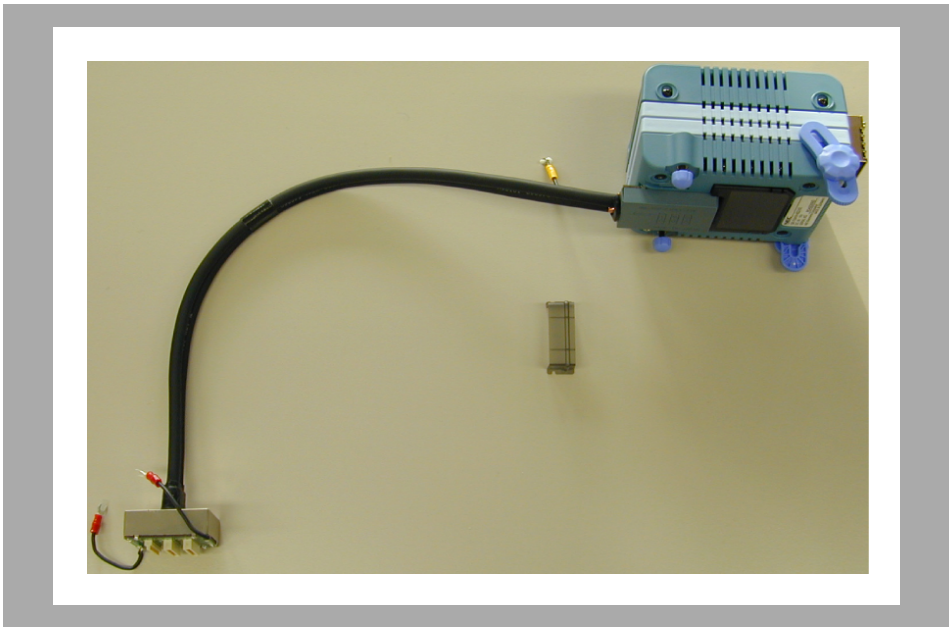


Figure 3-9 Connect IECUBE and the probe

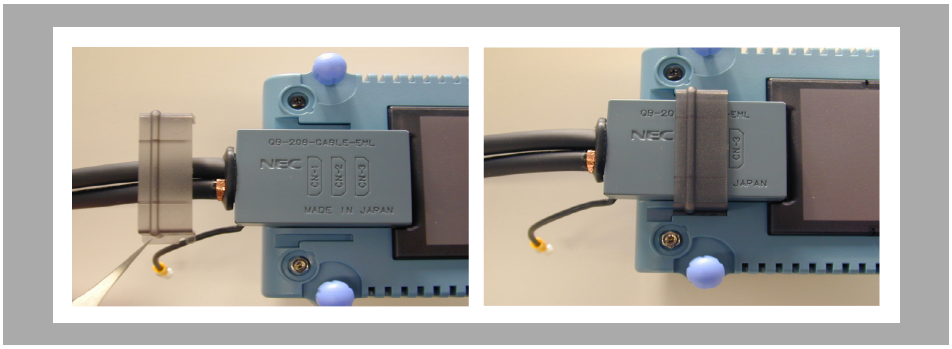


Figure 3-10 Insert Probe Holder in IECUBE

Insert the probe holder until it clicks (note direction).

(2) Connecting extension probe GND lines

The extension probe has three GND lines. Connect these lines to the IECUBE and the target system as follows:

<1> Fix a GND line of the extension probe to the nut on the bottom surface of IECUBE using a #0 or #1 precision cross-tip screwdriver.

<2> Insert the connector on the top of the extension probe in the connector at the bottom of the IECUBE from the lower side (note direction).

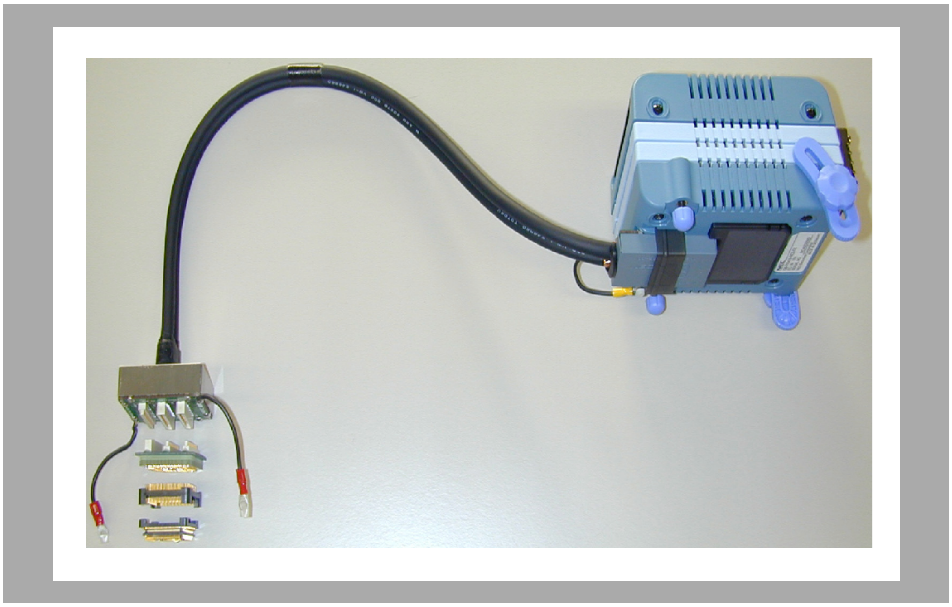


Figure 3-11 Connect of GND Lines

<3> Connect the exchange adapter and extension probe to the target connector.

<4> Connect two GND lines of the extension probe on the target system side to the GND block of the target system. If the pin or screw is fixed on the GND block of the target system, remove the transparent pin cover at the top of the GND line and fix the Y-branch pin of the GND line to the target system. In the same manner, if the GND pad on the target system is exposed, fix the Y-branch pin to the pad on the target system by soldering. (Recommended iron temperature: 300°C)

<5> If there is only one GND connector on the target system, connect one side and cut off the other GND lines using nippers, or leave as is without removing the pin cover.

<6> The length of the GND line shank (insulation block) is approximately 60 mm. As shown in Figure 2-10, at least one connectable GND is necessary within a radius of approximately 60 mm from the three locations on the extension probe to which the target system is connected. The GND lines on the emulation probe are soldered at the position of J and K in Figure 2-10. When soldering the GND line at the position of L or M, remove a GND line soldered at J or K and solder it at L or M.

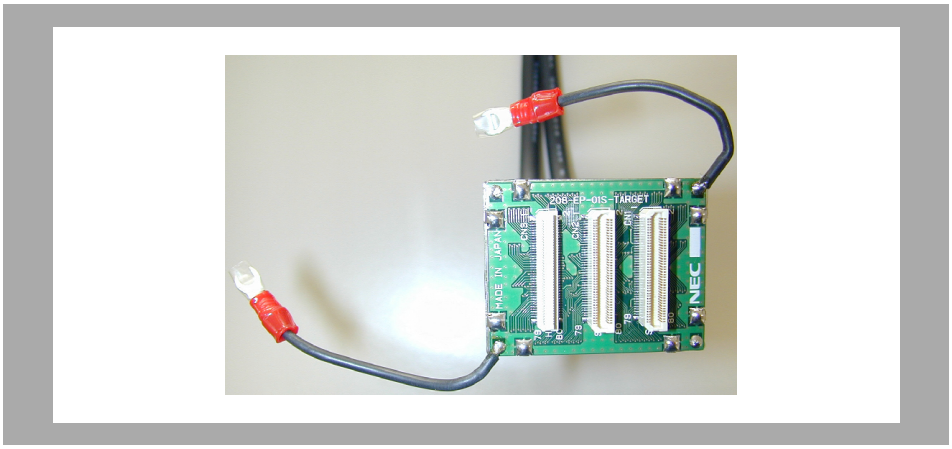


Figure 3-12 Location for Connecting GND Line

(3) Maintaining insulation

When the IECUBE and the target system are connected using the extension probe, adjust the height of IECUBE using the front spacer and rear spacer in order to maintain insulation with the target system.

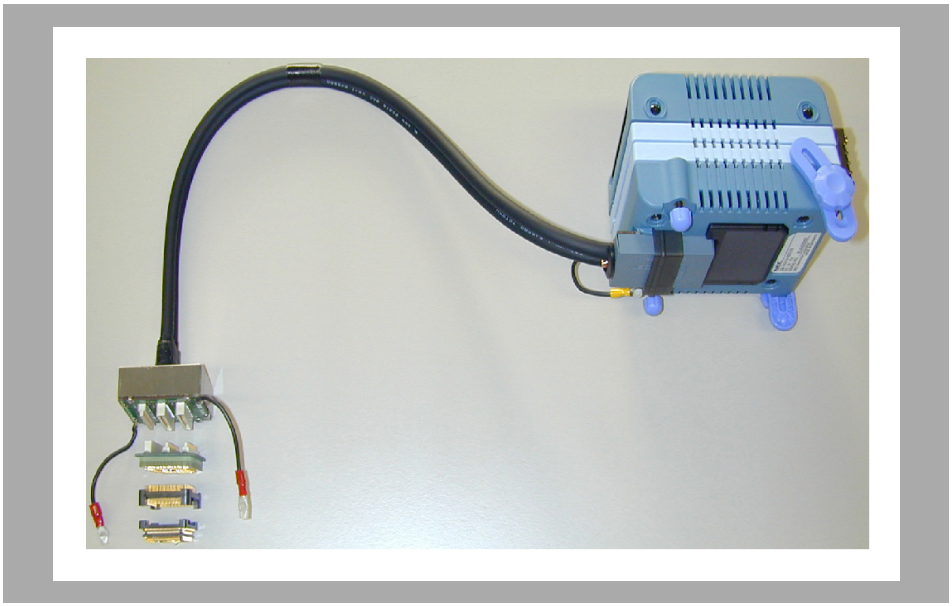


Figure 3-13 Connection when using Extension Probe

(4) Cautions on using extension probe

Note the following points when using the extension probe.

• Ensure that stress from the extension probe is not applied to the target connector. Hold the exchange adapter with your fingers when removing it so that no stress is applied to the target connector.

• Connect the GND line of the extension probe to the IECUBE and the target system; otherwise the impedance of the cable becomes unstable, which may cause degradation of the signal transmission characteristics or distortion of the output waveform with respect to the input waveform.

• If the external bus interface is used when the extension probe is used, increase the data wait by one. (Increase the value set to the DWC register by one.)

3.8 Connecting USB Interface Cable and AC Adapter

Connect the computer and the IECUBE using the USB interface cable supplied with IECUBE. Insert the power supply connector on the rear side of IECUBE and insert the AC adapter plug supplied with IECUBE in the outlet.

The AC adapter supports voltages from 100 V to 240 V by exchanging the AC plug. A 100 V AC plug is mounted at shipment. To use the IECUBE with 220 V or 240 V, exchange the AC plug for one that supports 220 V or 240 V (both included with IECUBE).

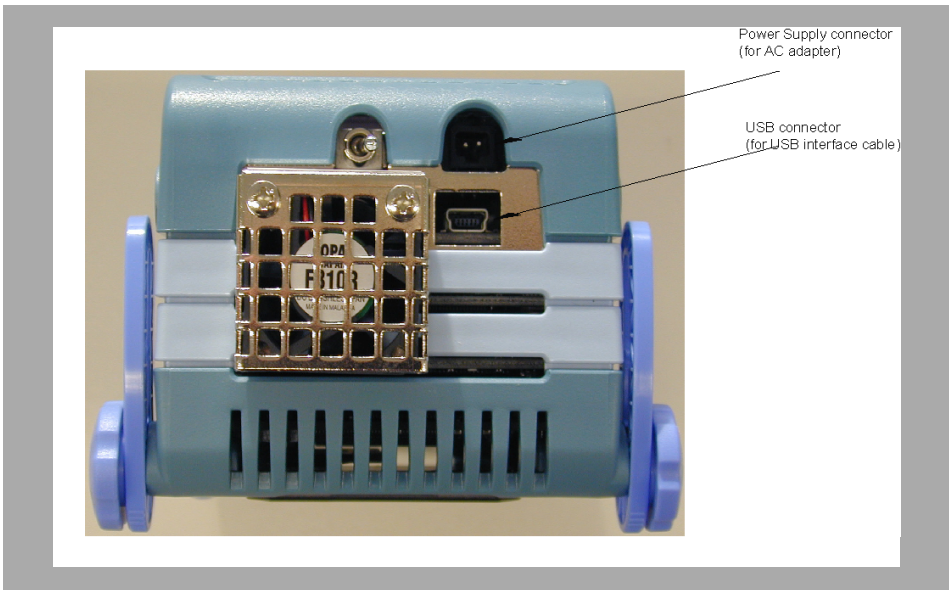


Figure 3-14 Connector Position

3.9 Power Application/Shutdown

Follow the sequence below when activating or terminating the emulator to avoid the risk of damaging the target system or the IECUBE.

- When activating the emulator:

Apply power to IECUBE -> Apply power to the target system[Note] -> Activates the debugger.

- When terminating the emulator:

Terminate the debugger -> Shut down power to the target system -> Shut down power to IECUBE.

Note This step is not required when the target system is not connected.

Chapter 4 List of Factory Setting

Table 4-1 SW1 DIP-switch factory settings

SW1-1	SW1-2	SW1-3	SW1-4	SW1-5	SW1-6	SW1-7	SW1-8
On	On			On	On		On
		Off	Off			Off	

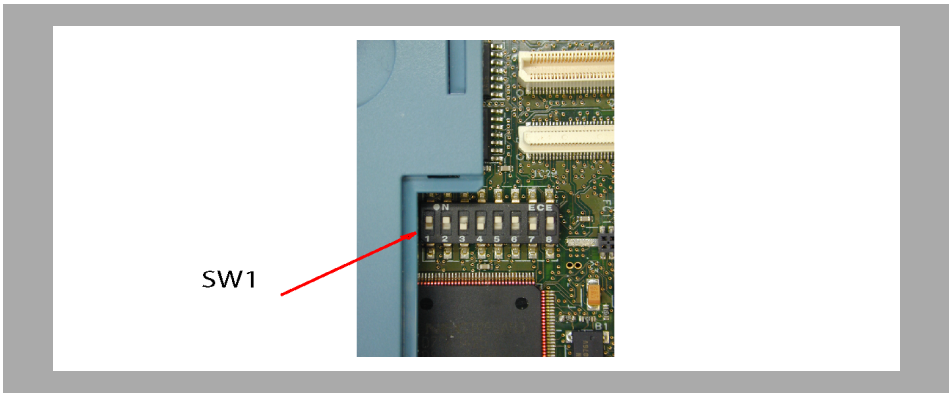


Figure 4-1 SW1 - factory setting

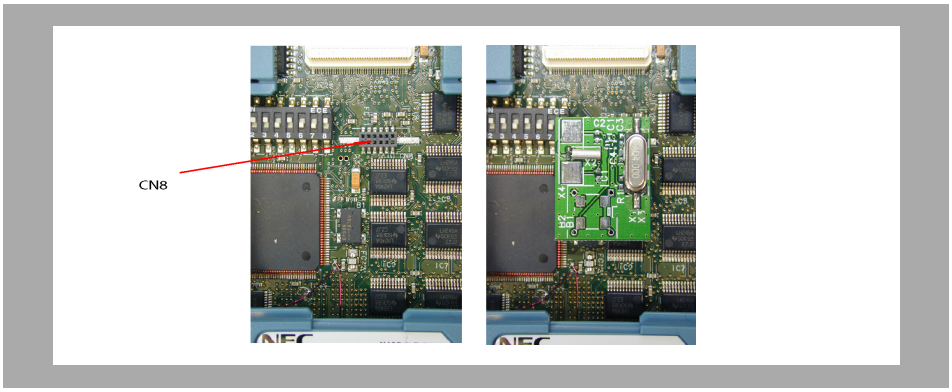


Figure 4-2 CN8 - mounted on CLK adapter board

Figure 4-3 Power supply switch factory settings off)

Chapter 5 Debug Function

The IECUBE is an In-Circuit-Emulator for the V850 32-bit RISC microcontroller family. It is used to emulate a V850E respectively V850ES core based microcontroller device. By using IECUBE, hardware and software can be debugged efficiently in system development. In this manual, the functionalities of the basic debugging features of IECUBE are described.

Table 5-1 Terminology

Term	Meaning
CPU	Central Processing Unit
EVA chip	CPU evaluation chip
PEC	Peripheral evaluation chip
RISC	Reduced Instruction Set Computer
EVA	Evaluation
DCU	Debug Control Unit
RRM	Real-Time RAM Monitor
MEMC	Memory Controller
INTC	Interrupt Controller
VSB	V850 System Bus
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
EXMEN	External mapped memory
iROM	internal ROM
iRAM	internal RAM
SFR	special function register
PC	Program Counter
PSW	Processor Status Word
FIFO	First in First out
I/F	Interface

5.1 Overview

A simplified block diagram of the IECUBE V850 In-Circuit-Emulator and its main functional components is shown in the figure below. The centre of IECUBE emulator is built by the V850E/ES based CPU EVA chip. The CPU EVA chip does realize the architecture and furthermore the functional emulation of the V850E respectively V850ES CPU core.

The V850 32-bit RISC microcontroller family offers a wide range of different peripherals and different peripheral sets. To cover the emulation of the corresponding peripherals, the V850 target device is used as peripheral EVA chip.

All debug specific functions like tracing, break control, time measurement and the Real-Time RAM monitor function is handled by the debug control logic of IECUBE.

The communication to the host computer and finally the interaction between IECUBE and the corresponding software debugger running on the host computer is realized by the control logic.

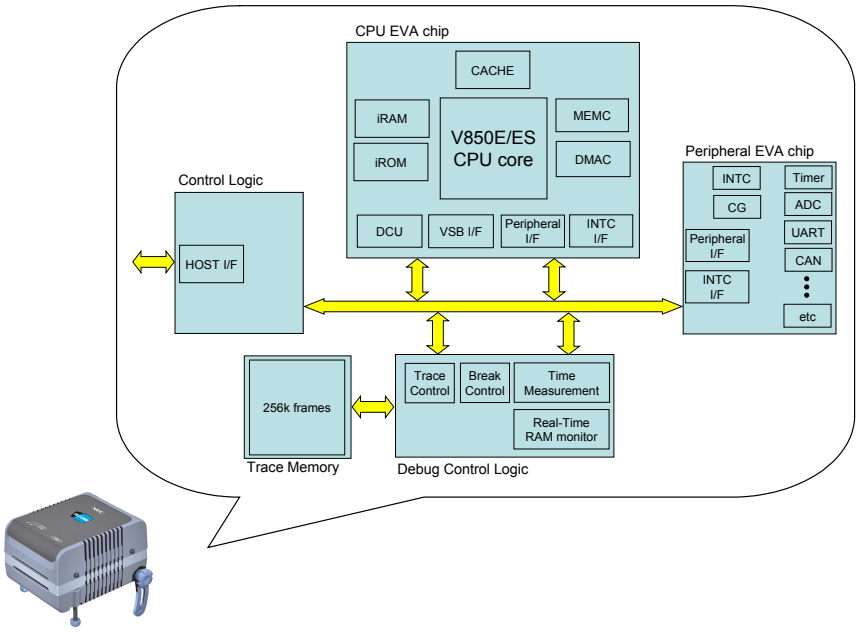


Figure 5-1 Simplified block diagram of IECUBE

5.2 Break functions

In principle we do distinguish between two kinds of break functions. First the break functions that are realized by the dedicated emulator hardware / debug control logic and second by the break functions that are realized by software emulation. This mechanism were used to implement the hardware respectively software breakpoint functionality.

Normally, the supported debuggers for the IECUBE emulator use software breakpoints as default configuration. Please check your debugger manual for information about the usage and configuration of hardware respectively software break functions.

5.2.1 Hardware breakpoints (Event detection break)

Function to stop the user program execution upon detection of a specified break event condition. The hardware breaks are managed by event conditions. In principle there are two different kinds of basic events supported, execution and access events. With regard to the execution events hardware breakpoints can be specified with the condition before execution or after execution. By specifying the before execution condition the two internal CPU core breakpoints are used. By specifying the after execution condition the hardware breakpoints of the event unit are used. To get more details on the different event conditions please refer to the chapter events of this document.

5.2.2 Software breakpoints

Function to replace the instruction at the specified address by a software break instruction (DBTRAP) and stopping the user program execution. Because software breakpoints are implemented by replacing the corresponding instruction the number of available software breaks is theoretically not limited. It depends on the used software debugger. Please refer to the documentation of your software debugger to get the information about the amount of supported software breakpoints.

Because the usage of software breakpoints does change the instruction on the specified address, please keep in mind that for error calculation on the program memory by using for instance a CRC algorithm, wrong results will be generated. To overcome this, the usage of hardware breakpoints is recommended.

The functionality of software breakpoints is limited. Software breakpoints can not be set to external ROM, also the definition of complex break conditions is not supported.

5.2.3 Fail-Safe break function

By using the Fail-Safe break function illegal memory accesses can be detected, for instance a read access to an unmapped memory address or a write access to read only SFR. In case of enabling the Fail-Safe break function the user program

execution is stopped immediately upon an illegal memory access is detected. The IECUBE emulator allows to specify the Fail-Safe break function for different memory regions, there are:

- External mapped memory (EXMEM)
- Internal ROM (iROM)
- Internal RAM (iRAM)
- SFR area

The Fail-Safe break functions for iROM, iRAM and EXMEM are realized by the CPU EVA chip of the IECUBE emulator. The SFR area protection is realized by the control logic. The initialization of CPU EVA chip respectively the control logic and the iROM, iRAM, EXMEM and SFR area protection regions is done based on the corresponding device file information loaded during debugger start-up phase.

Please refer to the documentation of your debugger, if the Fail-Safe function is enabled per default after start-up.

5.3 Events

Events do specify specific states of the target system during debugging. For instance, read or write accesses to a specific address or an address area, or code fetch from a specific address or an address area. This events can be used as action triggers for specific debugging function, such as hardware break, trace or time measurement.

There are two basic event types, the execution event and the access event. Additional one link event can be specified.

5.3.1 Execution events

The IECUBE emulator supports totally ten execution events. They are composed of two before execution events and eight after execution events.

5.3.1.1 Before execution event

The before executions events can be used exclusively as triggers for hardware breakpoints. No trace or time measurement function can be triggered. Additional the specification of an address range is not possible by using the before execution events. To realize the before execution events the two internal CPU core breakpoints are used. The necessity to use the internal core breakpoints comes from the fact, that the code execution has to be interrupted just within the instruction fetch stage of the five stage pipeline. This is not possible by using the event unit, because it is not an integrated part of the CPU core.

5.3.1.2 After execution event

The eight after execution events are realized by the event unit. These events can be used to trigger the complete debugging features of the IECUBE emulator, such as hardware breaks, trace and time measurement. Additional address ranges can be specified by the usage of an after execution event. In case an address range is specified only four execution events are available, because each address range is specified by a pair of two execution events.

5.3.2 Access events

The IECUBE emulator supports up to six access events. An access event can be used to detect write or read accesses to a single address or to an address range. In case an address range is specified only three access events are available, because each address range is specified by a pair of two access events.

5.3.3 Link event

By using a link event you can set a sequence of after execution or access events that must occur before a link event is triggered. A link event can trigger debugging functions like hardware breaks, trace or time measurement. The IECUBE allows to specify one link event in total.

5.4 Break / RUN mode

As basic features, the IECUBE emulator supports standard debug functions, for instances RUN, STOP, Single Step etc. Corresponding to this, the operation modes of the emulator can be divided into two basic modes, there are:

- **Break Mode:**

  Within this mode the program execution is stopped.

- **RUN Mode:**

  Within this mode the program execution is enabled

5.4.1 Break Mode

Within this mode the program execution is stopped. Direct access to the emulator hardware is allowed unrestricted. During the break mode the emulator configuration can be modified, break/event conditions can be set or changed, emulation memory can be read or written etc.

By entering the break mode caused by a forced break (STOP command) or triggered by a break condition, it is possible to stop dedicated peripherals by using the peripheral break function. Independent from this feature, if the emulation device does have a watchdog timer, the counter operation is stopped automatically by entering the break mode.

There are some characteristic features of break mode concerning DMA and interrupt operation. When the break mode is entered, for instance caused by a break, just in the same time a DMA transfer was issued, the DMA transfer is not interrupted. The DMA transfer will continue operation till it is finished.

5.4.2 RUN Mode

Within this mode the program execution is enabled. The program execution is performed in real-time. During this mode debug features like the Real-time RAM monitor function can be used without losing the real-time capability of the emulator.

Some debuggers do support additional features like standard I/O and also file I/O. This feature allows to redirect the standard I/O stream, for instance by using the “printf”, “getchar”, “putchar” library functions, to the I/O window of the debugger. To parse the standard I/O stream to the I/O window of the debugger, the emulator switches short-term to the break mode. Consequently no real-time program execution is preformed.

Additional when using the pseudo Real-time RAM monitor function no real-time program execution is performed. Please refer to the chapter “Pseudo Real-Time

RAM monitor function” of this document to get more details about this particular mode.

5.5 Trace

The V850 IECUBE offers a trace function that is used to save the history of user program execution like the corresponding instruction fetch or data access information to the trace memory. The IECUBE emulator supports a trace memory with a depth of 256k frames. Additional the trace function of IECUBE allows to store the history of DMA (direct memory accesses). Within this mode the start and end points of a DMA transfer are stored to the trace memory. However the source / destination addresses and the transferred data of a DMA are not stored to the trace memory.

The trace function of the IECUBE emulator is realized by the CPU emulation chip (EVA chip) that offers an N-Wire based trace interface and the debug control logic that collects and processes the raw trace packets generated by the EVA chip.

The IECUBE emulator supports different trace modes which allow to collect and observe individual sets of trace information. These trace modes can be configured separately but also a combination of different trace information is supported by the emulator. Please refer to the corresponding user's manual of your preferred debugger to get more details on the possible configuration. The following trace modes are supported by the IECUBE emulator:

- **BranchPC trace:**

  Traces only the PC information of an executed BRANCH instruction. By using this trace mode, the completion of the trace information is done by the debugger. This is done by inserting the instructions that have been executed between the BRANCHES within the trace window.

- **DataAccessPC trace:**

  Traces the PC information of the executed instruction that performs a memory access (data access).

- **InstructionPC trace:**

  Traces the PC information of each executed instruction.

- **DataAccess trace:**

  Traces only the data access address of the corresponding data access instruction.

- **Data trace:**

  Traces only the data written or read by the corresponding data access instruction.

The time stamp information of a trace frame is generated within the debug control logic. It can be based on the CPU clock or on the clock of the debug control logic (typical 50 MHz) that is unrelated to the actual CPU clock. By using the CPU clock as time reference the minimum resolution of a trace frame is four CPU clocks,

because one trace frame consist of four trace packages, whereby a trace package is generated each CPU clock. By using the external clock of the debug control logic the time stamp information does show the time required to read in a trace package from the EVA chip to the debug control logic. Also in this mode the trace package information is generated within the FIFO of the EVA chip and transferred to the debug control logic. Therefore the time stamp information generated within the debug control logic does not show the precise execution time of a single CPU instruction. It should mainly be used for time measurement of instruction sequences. To realize a CPU clock-cycle precise time measurement, please use the timer function of the IECUBE emulator.

In the following the principle operation of the trace function on the IECUBE emulator and the interaction between the CPU EVA chip and the debug control logic is described:

(1) In case the combination of BranchPC, DataAddress, Data and DataAccessPC trace is used at the same time up to four trace entries (of maximum 32) are stored at one CPU cycle within the FIFO.

(2) The debug control logic fetches per each single CPU clock cycle one trace entry from the FIFO.

(3) The fetched trace entry is stored within one of the temporary Buffers of the debug control logic.

(4) If the trace operation did just started, the trace entry is stored beginning with Buffer 0 and the time stamp counter is incremented by four.

(5) When all four trace entries are stored in the temporary buffers, one trace frame is build up together including the time stamp counter value and is stored to the trace memory. After the next writing to the temporary Buffer 0 the time stamp counter is incremented again by four.

As described above the resolution of the time stamp counter is minimum 4 CPU clocks, but just in that case that no FIFO overrun occurs or that the number of trace entries that were stored to the FIFO within each instruction cycle is less or equal then the required clock cycles of the instructions under current execution.

A FIFO overrun error can occur if the execution of the program becomes faster then the FIFO can be fetched from the debug control logic. For instance, when performing BranchPC, DataAddress, Data and DataAccessPC trace simultaneous up to four trace entries can be written to the FIFO within one CPU clock. On the other hand to fetch these trace information from the FIFO it needs four CPU clocks. Now let’s think about the following scenario. Within an application program a memory initialization is done in a loop. On each loop iteration four memory accesses were done. The program execution of a single loop needs 9 clocks, but the FIFO is loaded with 16 trace entries. Consequently, the debug control logic needs 16 CPU clocks to fetch the trace entries from the FIFO. After a few loop iterations a FIFO overrun occurs, because the program is running faster then the trace entries can be fetched from the FIFO. Remember, fetching one trace entry from the FIFO needs one CPU clock cycle. As consequence, trace data and time stamp information will be lost and the tracer starts a synchronization after erasing the FIFO. To avoid such a behavior, reduce the number of trace entries that are written to the FIFO within one CPU cycle by simple reducing the amount of different trace information that are collected simultaneous.

5.6 Real-Time RAM monitor function

This function offers the possibility to display variables or data allocated to a specified memory area in real-time. The Real-Time RAM monitor function is realized by the hardware (EVA chip and debug control logic) and does not change the real-time behaviour of the emulation system. Up to eight real-time areas can be specified, with a size of 256 bytes each. The RRM function and the trace function can not be used simultaneous. This is caused be the fact, that the RRM function uses the debug control unit (DCU) of the EVA chip exclusively to generate DataAccess and Data information.

5.7 Pseudo Real-Time RAM monitor function

Different to the Real-Time RAM function realized by the hardware of the IECUBE emulator, the pseudo Real-Time RAM monitor function is a pure software emulation. By using this function, the application is interrupted for a short time to read the corresponding variables or data that is part of the observation. This is done by running a background monitor program during break mode, to up-load the memory contents to the debugger. Consequently no real-time execution is supported when using the pseudo real-time RAM monitor function.

5.8 Timer

The timer function of the IECUBE offers two possible operations, the run-break timer function and the event timer function. The timer function is based on either a 50 MHz clock - making each time unit 20 nanoseconds (ns) long - or the external CPU clock.

5.8.1 Execution timer (run-break timer)

The timer function measures the execution time (run-break time) from the start of user program execution until a break.

5.8.2 Event timers

The timer function can be used to measures the execution time in a specific user program interval by using timer events. Please note, the before execution events can not be used to trigger the timer function. Up to seven timer events are supported by the IECUBE emulator.

5.9 Peripheral Break function

This function of the IECUBE emulator allows to specify, if dedicated peripherals of the emulation device should be stopped simultaneously when the user program execution is stopped caused by a SW/HW breakpoint or stopped manually by a user request. The peripheral break function must be supported by the corresponding macro of the emulation device. Typically it is supported by Timer, Watch-Timer or A/D converter macros. Please refer to the corresponding chapter of this documentation to get information if this function is supported for the emulated V850 device.

5.10 Limitations

5.10.1 Break mode

Within the break mode it might happen that interrupts of an interrupt source were lost, if the interrupt is requested several times. Please note that only one interrupt request is detected for one interrupt source. If the same interrupts occurs several times during break mode only one interrupt acknowledgment is done.

5.10.2 ROM correction function

The ROM correction, if supported by the target device, can not be emulated by the IECUBE. The IECUBE supplies “dummy” register’s as space holder for the ROM correction SFR’s only. Additional, the access timings of these registers are not identical to these of the real device.

5.10.3 DBTRAP instruction

The DBTRAP instruction can not be used in combination with the IECUBE: The DBTRAP instruction is reserved exclusively for debugging purpose.

5.10.4 DBPC, DBPSW, and ECR registers

The DBPC, DBPSW, and ECR registers cannot be accessed during break mode. If a value is written to any of these registers during break mode, the written value is ignored.

5.10.5 Illegal Opcode detection (ILLGOP)

When enabling the illegal opcode detection function, the IECUBE can detect if an illegal opcode was fetched during program execution. In case of ILLGOP detection, the program jumps automatically to the corresponding exception trap at address 0x60 and continues execution of the exception trap handler. At this time the corresponding address of illegal opcode detection is displayed by the debugger.

However the DBPC and DBPSW registers are reserved exclusively for debugging purpose. This causes that the registers were not loaded with the correct return address and current PSW value by jumping to the exception trap vector. Consequently, when leaving the exception trap handler by using the DBRET instruction incorrect values are restored to the PC and PSW registers and a program crash may happen. Therefore, when using the illegal opcode detection, please leave the exception trap handler for instance by doing a software reset.

Note, the implementation of the ILLGOP detection function must be supported by the selected debugger and might be different. Please check your debugger manual for information about the usage and configuration of the ILLGOP detection function.

Chapter 6 Differences Between Target Device and IECUBE

This chapter explains the differences between the IECUBE to the V850E/Dx3 devices.

Three user's manuals cover the description of V850E/Dx3 devices, the core architecture (V850E1) and the IECUBE.

Refer to these or later version:

- **Device:**

<1> Hardware UM -V850E/Dx3 U17566EE1V0UM00 or - current version

<2> Architecture UM -V850E/Dx3 U14559EJ3V1UM00 or - current version

<3> Operating Precautions CN -V850E/Dx3 Dx3 OP's or - current version

- **QB-703427:**

<4> Operating Precautions CN -QB-703427 EEDT-OP-0030-6.0 or - current version

<5> IECUBE UM (this manual) -QB-703427 U18350EE1V2UM00 or - current version

Note Download the documents from the NEC Electronics (Europe) GmbH web site.

URL: http://www.eu.necel.com/products/micro/

6.1 Functions Emulated Differently

The following functions are emulated differently on the IECUBE and V850E/Dx3 device:

- POC - emulated by a discrete circuit

- Interface Pins - emulated by a different logic circuit than on device

- RESET - emulated by discrete circuit

- MODE switch - not used and not emulated

- IRAM behaviour - the contents after power on might differ

- FOUT- and WDT-clock standby mode - different emulation circuit

- PSM.OSCDIS - different emulation

- Oscillation stabilization time - emulated by different clk circuit

- ADC break precaution - emulation caution

- Port register $\overline{\text{RESET}}$ value- emulated by a different logic circuit than on device

- CSI/UART Caution - emulation caution

- Power consumption - emulation caution

- SSCG precaution - emulation caution

- MEMC interface - emulation timing caution

- internal low-speed on-chip oscillator frequency - fixed frequency

6.1.1 Differences when emulating POC

The target VDD power is checked against the POC threshold voltage via a comparator device, set to 3.35 V value typical (3.2 V min. and 3.5 V max.). This detection information is used for emulating the POC function.

6.1.2 Differences on Interface Pins (Electrical Characteristics)

The electrical characteristic of the pins that are emulated differently than the pins on the devices: µPD70F342x (x = 1/2/3/4/5), µPD70F3427 and µPD70(F)341x (x = 6/7).

<1> Ports alpha - emulated target interface via Level shifter buffer NOTE

<2> Ports numeric - no difference

<3> Clock - XT1, XT2, X1, X2 not used Note 2

<4> Power

- internal supply in standalone mode (no target connected) Note 3

- External supply on AVDD, BVDD, DVDD, MVDD, SMVDD, VDD

(on 144-pin device, the MVDD is supplied internally) / In both modes, the power consumption to the device might differ.

<5> Ground - a common ground used

- One ground pin (VSS50/pin63) is used for target connection detection. On this pin an internal 10 kΩ pull up is used.

<6> RESET - A 47 kΩ pull down is used. The input characteristic differs.

<7> FLMD0 - Not used. The input characteristic differs.

<8> REGCx - A 4.7 µF and a 100 nF capacitor are used in parallel. The input characteristic differs.

The electrical characteristic of all signals are different than on µPD70F342x (x = 1/2/3/4/5), µPD70F3427 and µPD70(F)341x (x = 6/7). This is based on the PCB wiring, all connectors and the target probe wiring. These results in an additional load on all pins and therefore the characteristic and the timing of the signals are affected. This should be taken into consideration when using the IECUBE on the target board.

The current in the standby modes can not be emulated.

Note 1. The electrical characteristic of the µPD7885/NEC level shifter :
Voltage range: 1.8 V to 5.5 V CMOS level / Propagation delay typ. 10 ns / Output current IOH- 10 mA (group of 6 outputs- 30 mA) / Output current IOL 20 mA (group

of 6 outputs 35 mA) / Note this different characteristic when using the MVDD Port group.
2. Using crystals on target is not supported.
3. The power structure of the IECUBE is as following
Using the IECUBE in standalone mode, the power is supplied internally fixed to 5.0 V for all power pins and AVREF. In standalone mode, no target system connection is allowed. Using the IECUBE in target mode, the availability of the target will be checked by VSS50/pin63. The pin is connected internally to a 10 kΩ pull-up. The target board provides a ground connection at this pin. This is the indication, that the target system is available. All power pins are connected directly to the power pins of the emulation device via an analogue switch.

Table 6-1 Power structure

VSS50/pin63	Power used from	Mode	Power checked on	AVDD/AVREF	AVDD/AVREF
Target connection	Target/internal	SW1 switch1	> 2.7 V	SW1 switch2	Power used from
open	internal	-	-	-	-
GND	target	on	all power pins	on	target
				off	internal
		off	on VDD pins onlyNote	on	target
				off	internal

All other power pins must supply a voltage in the defined range.

6.1.3 Differences when emulating RESET

The $\overline{\text{RESET}}$ function is emulated by using different input buffer. The input characteristic differ. An 47 kΩ pull down resistor is connected. Note this on target interface. The timing might be different.

6.1.4 Differences when emulating MODE switch

This input level is not checked by the IECUBE.

6.1.5 Break precaution related to ADC macro

(1) Details:

The following behaviour is valid for the IECUBE emulator only in case the peripheral break mode is active for the ADC macro:

<1> In case the peripheral break signal (SVSTOP = 1) is set while or after the conversion control bit ADA0CE has been set, the AD conversion is not started and the concerned interrupt INTAD will not be generated. Moreover the AD conversion will not start even if the break mode has been exited and the debugger operates in RUN mode.If the ADA0CE bit is set again during normal RUN mode without issuing the peripheral break signal, the ADC will operate as specified.

The peripheral break signal is issued under the following conditions:

a.) If one of these break is executed on the AD0ACE bit write instruction

Software break

Before-execution hardware break

After-execution hardware break

b.) If one of these break is executed on the first instruction following the AD0ACE bit write instruction

Software break

Before-execution hardware break

c.) If the following break is executed on the second instruction following the AD0ACE bit write instruction

Software break

<2> In case the peripheral break mode (SVSTOP=1) has been configured and the debugger operates in the debug (break-) mode, a write operation to the ADC concerned registers: ADA0M0, (ADA0M1(#)), ADA0M2, ADA0S, ADA0PFT, ADAPFM (#) when ADA0CE=1, the re-write of ADA0M1 is prohibited and will not cause the start of the ADC's reconversion. It makes no difference whether the write operations to the above mentioned ADC registers are executed via the debugger itself or via DMA that is not stopped when entering the break mode. Both write operations will cause the limitation.

(2) Workaround:

<1> When a software break is executed in case the peripheral break mode has been configured for the ADC macro, do not set the software break for the instruction the ADA0CE bit or at either of the following two instructions:

Example:

set1 7, ADA0M0 --- software break is prohibited

nop --- software break is prohibited

nop --- software break is prohibited

nop --- software break is possible to set from here on

<2> When a “before-execution hardware break” is executed and the peripheral break mode has been configured for the ADC macro, do not set the breakpoint at the same instruction that sets the ADA0CE bit or at either of the following instructions:

Example:

set1 7, ADA0M0 --- before-execution hardware break is prohibited

nop --- before-execution hardware break is prohibited

nop --- before-execution hardware break is possible to set from here on

<3> When a “before-execution hardware break” is executed in peripheral break mode and the peripheral break mode has been configured for the ADC macro, do not set the breakpoint at the same instruction that sets the ADA0CE bit:

Example:

set1 7, ADA0M0 --- after-execution hardware break is prohibited

nop --- after-execution hardware break is possible to set from here on

<4> If you want to proceed the write operation for the AD related registers during BREAK (debugger operates within the break mode), do not use peripheral break mode.

<5> If you want to proceed the DMA transfer which has AD related registers set as source/ destination for this DMA transfer <<<, do not use peripheral break mode.

Note In case a condition mentioned under Workarounds: 1, 2 or 3 will occur when setting one of the concerned breakpoints on the location of an interrupt-vector, no limitation will become valid due to the clock-cycles that are requested for the interrupt-response time!

6.1.6 iRAM

The content of the iRAM are unchanged after a $\overline{\text{RESET}}$ / target power off/on.

This behavior may differ from the real device. In the real device the contents of the iRAM may change to that contents before a $\overline{\text{RESET}}$ / power off/on.

6.1.7 FOUT- and WDT-clock supply differ from device in standby mode

(1) Details:

In standby mode, the FOUT clock supply (internal low-speed on-chip oscillator clock, if ROSTP=1 is set) will not stop.

In standby mode, the FOUT clock supply (internal low-speed on-chip oscillator clock, if SOSTP=1 is set) will not stop.

In standby mode, the WDT clock supply (internal low-speed on-chip oscillator clock, if ROSTP=1 is set) will not stop.

In standby mode, the WDT clock supply (internal low-speed on-chip oscillator clock, if SOSTP=1 is set) will not stop.

(2) Workaround:

None

6.1.8 PSM.OSCDIS reset value different to device

(1) Details:

The reset value of the OSCDIS is ‘1’. On real chip OSCDIS is set to ‘0’ during firmware execution.

(2) Workaround:

Initialize the OSCDIS after RESET or use the functions of the Debugger to initialize the OSCDIS before program start.

6.1.9 Timing different to device for oscillation stabilization time

(1) Details:

The oscillation stabilization time indicated by OSCSTAT differs to that of the real device. In emulation mode the oscillator runs permanently, so the time for oscillation start is not the same. After reset, because of the different OSCDIS setting, the oscillation stabilization counter start is different.

(2) Workaround:

None

6.1.10 Differences on Port register RESET value

(1) Details:

The reset value of the register described below differ.

SFR	Real-chip value	Emulator value	Action	Effects due to differing reset value
PICC3	0xF3	0xFF	Reset value	None
PICC8	0x3F	0xFF	Reset value	None
PICC10	0x0F	0xFF	Reset value	None

(2) Workaround:

Initialise the register after RESET or use the functions of the Debugger to initialize the register before program start.

6.1.11 CSI/DMA caution

CSI Caution:

When the CSIB continues to operate in debugger break-mode it may produce an overflow error as it continues to receive data during break-mode.

As a result, the overflow error flag CBnSTR.CBnOVE may be set and the reception error interrupt INTCBnRE may be generated. Any subsequent data received data are discarded. Note that the reception error interrupt INTCBnRE will not be serviced during break-mode, but on resumption of run-mode, provided the interrupt controller is configured accordingly.

The CSIB continues to operate during debugger break-mode

- in continuous reception mode
- in slave reception mode

If the DMA controller is used to fetch received data from the CBnRX register, the DMA controller continues to do so in debugger break-mode, but the data in CBnRX is not tagged as already read. Thus the next receive data during break-mode will generate an overflow error, as described above. If the specified number of data units in the DBCn register has been fetched from the CSIB the DCHCn.TCn is set and the DMA completion interrupt INTDMAn is generated. The INTDMAn request is serviced on resumption of resuming run-mode.

Since the DMA trigger interrupt INTCBnR is not generated in the event of an overflow (the INTCBnRE interrupt is generated instead), no further DMA triggers occur. Only the first received data in break-mode is transferred by the DMA to the RAM, all following data are discarded.

UART Caution:

When the UARTA continues to operate in debugger break-mode it may produce an overflow error as it continues to receive data during break-mode.

As a result, the overflow error flag UAnSTR.UAnOVE may be set and the reception error interrupt INTUAnRE may be generated. Any subsequent data received are discarded.

Note that the reception error interrupt INTUAnRE will not be serviced during break-mode, but on resumption of run-mode, provided the interrupt controller is configured accordingly.

If the DMA controller is used to fetch received data from the UAnRX register, the DMA controller continues to do so in debugger break-mode, but the data in UAnRX is not tagged as already read. Thus the next receive data during break-mode will generate an overflow error, as described above. If the specified number of data units in the DBCn register has been fetched from the UARTA the DCHCn.TCn is set and the DMA completion interrupt INTDMAn is generated. The INTDMAn request is serviced on resumption of run-mode.

Since the DMA trigger interrupt INTUAnR is not generated in the event of an overflow (the INTUAnRE interrupt is generated instead), no further DMA triggers occur. Only the first received data in break-mode is transferred by the DMA to the RAM, all following data are discarded.

6.1.12 Difference on power consumption

(1) Details:

Power consumption emulation cannot be used for the IECUBE. Internal clock supply is not stopped during the standby mode. Power consumption of the IECUBE is therefore different than of the target device.

(2) Workaround:

None

6.1.13 SSCG modulation setting is limited

Caution The change of the default setting may result in non error-free operation.

(1) Details:

In case of operation with 64 MHz, the modulation setting is ignored.

(2) Workaround:

With FPGA data bitstream V4.04 the following feature is supported:

SW1.7 = ON: Dithering is active if software enabled it, but error-free operation is not guaranteed.

SW1.7 = OFF: Dithering is disabled (at 64 MHz CPU frequency only), even if software enabled it.

6.1.14 Timing different to device for MEMC interface

(1) Details:

The QB703427 (with target probe) timing differ / is slower than the device timing. This is valid only for emulation mode selected for target device uPD70F3427.

(2) Workaround:

The need for additional wait state depends on the used frequency and external memory and must be checked against the timing values in each case. See timing specification tables below.

Note The probe QB-208-EP-01S propagation delay time is about 5ns. For target device timing specification of uPD70F3427 refer to document EASE-SP-8007 (current version).

Table 6-2 External Memory Access Asynchronous Read Timing

Parameter	Symbol	NAME	Cond.	Min	Max	Unit
Data input set up time D0-15 (vs.address)	<10>	TSAID			(2.0+WT)T-41	[ns]
Data input set up time D0-15 (vs.RD..)	<11>	TSRDID			(1.5+WD)T-37,5	[ns]
Data input set up time D16-31 (vs.address)	<10>	TSAID			(2.0+WT)T-41	[ns]
Data input set up time D16-31 (vs.RD..)	<11>	TSRDID			(1.5+WD)T-37,5	[ns]
RD Low level width	<12>	TWRDL		(1.5+WD)T-5		[ns]
RD High level width	<13>	TWRDH		(1.5+WAS+i)T-5		[ns]
Address to RD delay time	<14>	TDARD		(0.5+WAS)T-3,5		[ns]
CSn to RD delay time	<14a>	TDCRD		(0.5+WAS)T-3,5		[ns]
RD address delay time	<15>	TDRDA		iT+3,5		[ns]
RD address delay time (delayed RD)	<15a>	TDRDC		(-0.5+i)T-4,5		[ns]
Data input hold time (vs. RD..)	<16>	THRDID		-6		[ns]
Write data output delay time after RD..	<17>	TDRDOD		iT+5		[ns]

Table 6-3 External Memory Access Asynchronous Write Timing

Parameter	Symbol	NAME	Cond.	Min	Max	Unit
Address, WR delay time	<20>	TDAWR		(0.5+WAS)T-0		[ns]
CSn, WR delay time	<20a>	TDAWR		(0.5+WAS)T-0		[ns]
Address set up (vs. WR..)	<21>	TSAWR		(1.5+WT)T-0		[ns]
WR address delay time	<22>	TDWRA		(0.5+i)T-0		[ns]
WR CSn delay time	<22a>	TDWRA		(0.5+i)T-0		[ns]
WR High level width	<23>	TWWRH		(1+i+WAS)T-0		[ns]
WR Low level width	<24>	TWWRL		(1+WD)T-0		[ns]
Data output set up time D0-15 (vs. WR..)	<25>	TSODWR		(1+WT)T-10		[ns]
Data output hold time D0-15 (vs. WR..)	<26>	THWROD		(0.5+i)T-8,5		[ns]
Data output set up time D16-31 (vs. WR..)	<25>	TSODWR		(1+WT)T-10		[ns]
Data output hold time D16-31 (vs. WR..)	<26>	THWROD		(0.5+i)T-8,5		[ns]

Table 6-4 External Memory Access Synchronous Read Timing

Parameter	Symbol	NAME	Cond.	Min	Max	Unit
BCLK cycle time	<90>	TBCYC				[ns]
BCLK low	<91>	TBH				[ns]
BCLK high	<92>	TBL				[ns]
A0-23 Hold time from BCLK..	<93>	THAD		16		[ns]
A0-23 Setup time to BCLK..	<94>	THSAD		7,5		[ns]
CSx Hold time from BCLK..	<95>	THCS		16		[ns]
CSx Setup time to BCLK..	<96>	TSCS		7,5		[ns]
RD.. after BLCK.. Hold	<97>	THRDF		16		[ns]
RD.. to BLCK.. setup	<98>	TSRDF		11		[ns]
RD.. after BLCK.. Hold	<99>	THRDR		16		[ns]
RD..to BLCK.. setup	<100>	TSRDR		11		[ns]
Delayed RD.. after BLCK..Hold	<99a>	THRDRD		19		[ns]
Delayed RD.. to BLCK.. setup	<100a>	TSRDRD		3		[ns]
Data input set up time D0-15 (vs. BCLK..)	<101>	TSRDID		32,5		[ns]
Data input set up time D16-31 (vs. BCLK..)	<101a>	TSRDIDa		32,5		[ns]
Data input hold time D0-15 (vs. BCLK..)	<102>	THRDID		-12		[ns]
Data input hold time D16-31 (vs. BCLK..)	<102a>	THRDIDa		-12		[ns]
WAIT input set up time a (vs. BCLK..)	<103>	TSWK		32,5		[ns]
WAIT input hold time (vs. BCLK..)	<104>	THWK		-12		[ns]

Table 6-5 External Memory Access Synchronous Write Timing

Parameter	Symbol	NAME	Cond.	Min	Max	Unit
BCLK cycle time	<110>	TBCYC				[ns]
BCLK low	<111>	TBH				[ns]
BCLK high	<112>	TBL				[ns]
A0-23 Hold time from BCLK..	<113>	THAD		16		[ns]
A0-23 Setup time to BCLK..	<114>	THSAD		7,5		[ns]
CSx Hold time from BCLK..	<115>	THCS		16		[ns]
CSx Setup time to BCLK..	<116>	TSCS		7,5		[ns]
WR.. after BLCK.. Hold	<117>	THWRF		16		[ns]
WR.. to BLCK.. setup	<118>	TSWRF		7,5		[ns]
WR.. after BLCK..Hold	<119>	THWRR		16		[ns]
WR.. to BLCK.. setup	<120>	TSWRR		7,5		[ns]
Data output set up time D0-15 (vs. BCLK..)	<121>	TSWDO			10	[ns]
Data output set up time D16-31 (vs. BCLK..)	<121a>	TSWDOa			10	[ns]
Data output hold time D0-15 (vs. BCLK..)	<122>	THWDO			10	[ns]
Data output hold time D16-31 (vs. BCLK..)	<122a>	THWDOa			10	[ns]
WAIT input set up timea (vs. BCLK..)	<123>	TSWK		32,5		[ns]
WAIT input hold time (vs. BCLK..)	<124>	THWK		-12		[ns]

6.1.15 Internal low-speed on-chip oscillator fixed to 200 kHz

(1) Details:

The internal low-speed on-chip oscillator frequency may differ from the device to be emulated.

(2) Workaround:

None

6.2 Notes on Emulation

The following listed notes apply when using IECUBE.

<1> register - access in break mode

<2> register - access if software break is set

<3> trace - display order

<4> instruction ex. - execution of instructions if hardware break is set

<5> memory area - non map break

<6> instruction ex. - guarded area access break delay

<7> internal RAM - do not use simultaneous DMA transfer

<8> Messages on starting Debugger - description

6.2.1 Access to CPU register DBPSW, DBPC, ECR.

DBPSW, DBPC and ECR cannot be accessed in break mode. If written, the value is discarded, if read 0 is always read.

6.2.2 PSC register access

(1) Details:

The debugger hangs up if a software break is set at the NOP instruction immediately after the PSC register is accessed.

(2) Workaround:

Example:

MOV 0x2, R1

ST.B R1, PRCMD

ST.B R1, PSC

NOP The debugger hangs up if a software break is set here

NOP Setting a software break here on causes no problem

Use a hardware break to set a break immediately after the PSC register is accessed.

6.2.3 Trace display order of data access trace

(1) Details:

When the trace mode - Data access trace - is enabled to display the access history the displayed order may be reversed in the following cases:

(a) When a write instruction follows a read instruction.

(b) When a bit manipulation instruction that performs a read/modify/write operation

(SET1, NOT1, CLR1) is performed

The trace result is displayed in the following order: 1st: write, 2nd: read.

The reversed trace display order is related to the read data accesses only, because instructions and data accesses are output by the DCU separately and each information belongs to different trace packages. With a “Store” instruction the corresponding data are available immediately, with a “Load” instruction the data access has to be executed first to get the corresponding data.

(2) Workaround:

Use TraceMode: BranchPC + DataAddress + Data + DataAccessPC. In this case the trace data can be classified correctly, because also DataAccessPC is available.

6.2.4 Simultaneous execution of two instructions when hardware break is set

(1) Details:

Suppose that two instructions “instruction A” followed by an “instruction B” are executed simultaneously. The execution result of these instructions when a hardware break is set is shown in the table below. As a result the break may occur at a different location from the set address, or the break may even not occur. The meaning of simultaneously is as follows: Two consecutive instructions are processed at the same time inside the pipeline (at different stages). A real simultaneous execution is not possible in the V850E core.

Table 6-6 Break Timing

No	Instruction A	Instruction B	Execution Result
1	Break before execution	No break setting	Break before execution of A
2	Break after execution	No break setting	Break after execution of B
3	No break setting	Break before execution	Break before execution of A
4	No break setting	Break after execution	Break after execution of B
5	Break before execution	Break before execution	Break before execution of A
6	Break before execution	Break after execution	Break before execution of A and after execution of B
7	Break after execution	Break before execution	Break before execution of A and after execution of B
8	Break after execution	Break after execution	Break after execution of B

(2) Workaround:

To avoid the above behaviour set a software break instead of a hardware break.

6.2.5 Non map break

(1) Details:

Normally a non map break occurs, if a program fetch is performed on an unused memory area in the emulator. The non map break is, however, not generated in the top 16 bytes of unused areas

(2) Workaround:

None

6.2.6 Guarded area access break delay

(1) Details:

When a guarded area is entered by a jump instruction or an illegal sfr access is detected, the guard access break is delayed for approximately 5 instruction cycles. The exact cycles required for this detection is not determinable due to the internal detection structure of the IECUBE.

(2) Workaround:

None

6.2.7 Break during program execution in internal RAM.

(1) Details:

An unexpected break may occur when a peripheral I/O register is accessed during program execution from internal RAM.

(2) Workaround:

Cancel the fail-save break setting for the internal RAM in the debugger.

When using Green Hills Multi: Cancel the fail-save break for “ramgrd” and “ramgrdv” using the Target command “flsf”.

6.2.8 Program execution in internal RAM and simultaneous DMA transfer access to/from internal RAM.

Note This behaviour does only occur on device, not on IECUBE.

(1) Details:

When executing one of the following instructions, located in the internal RAM, do not execute a DMA transfer that transfers data to/from the internal RAM (transfer source/destination), because the CPU may not operate correctly afterwards:

A bit manipulation instruction (SET1, CLR1 or NOT1)

(2) Unaffected cases

The critical situation does not occur if no instruction is executed in the internal RAM, or no DMA transfer is performed on the internal RAM.

(3) Workaround:

Implement any of the following workarounds:

Workaround

- Do not perform DMA transfers to the internal RAM when these instructions are executed from internal RAM.

6.3 Functions Not Supported

<1> ROM correction[Note]

<2> Flash self programming

<3> N-Wire

Note The ROM correction function is not supported. The IECUBE supplies “dummy” register’s as space holder for the ROM correction SFR’s only. The access time of these registers are not identical to those of the real device.

6.4 Notes on Debug functions

The following notes apply to debugger functions that are used in IECUBE break mode. This may result in not real time behaviour of the debug session. (See the users manual of the debugger for the description of the function).

6.4.1 Debugger functions using break mode

The program execution is stopped in break mode and restarted. This may result in non-real-time behaviour of the debug session. All activities during break mode (for example interrupt execution, DMA activity) may differ to device execution.

Following functions using break mode:

- Viewing and modifying variables with the Data Explorer. Forcing an update by selecting View -> Refresh Views or by issuing the update command, the break mode of IECUBE will be used.
- Stop of SW execution (Breakpoints)
- I/O functions (printf / scanf)

6.4.2 Macro supporting peripheral break function

The Peripheral Break function is supported by the following macro's:

- A/D Converter (ADC)
- Timer P (TMP0 to TMP3)
- Timer Z (TMZ0 to TMZ9)

Chapter 7 Notes on Target System Design

This chapter explains notes on target system design, including areas in which devices should not be mounted on the target system and areas with a height restriction for devices to be mounted. These values may change depending on whether a space adapter or target probe is used.

For the required target probe connectors, for 208-pins QFP, 144-pins LQFP and 100-pins LQFP, refer to the list below:

The following package type is used:

- µPD70F3427GD(A)-LML-AX: 208QFP 28mm × 28mm / 0.5mm
- µPD70F342xGJ(A)-GAE-AX: 144LQFP 20mm × 20mm / 0.5mm
- uPD70(F)341xGC(A)-UEU-QS-AX: 100LQFP 14mm × 14mm / 0.5mm

All drawings views are from top.

7.1 When Extension Probe Is Not Used

7.1.1 Area dimensions - 100-pin

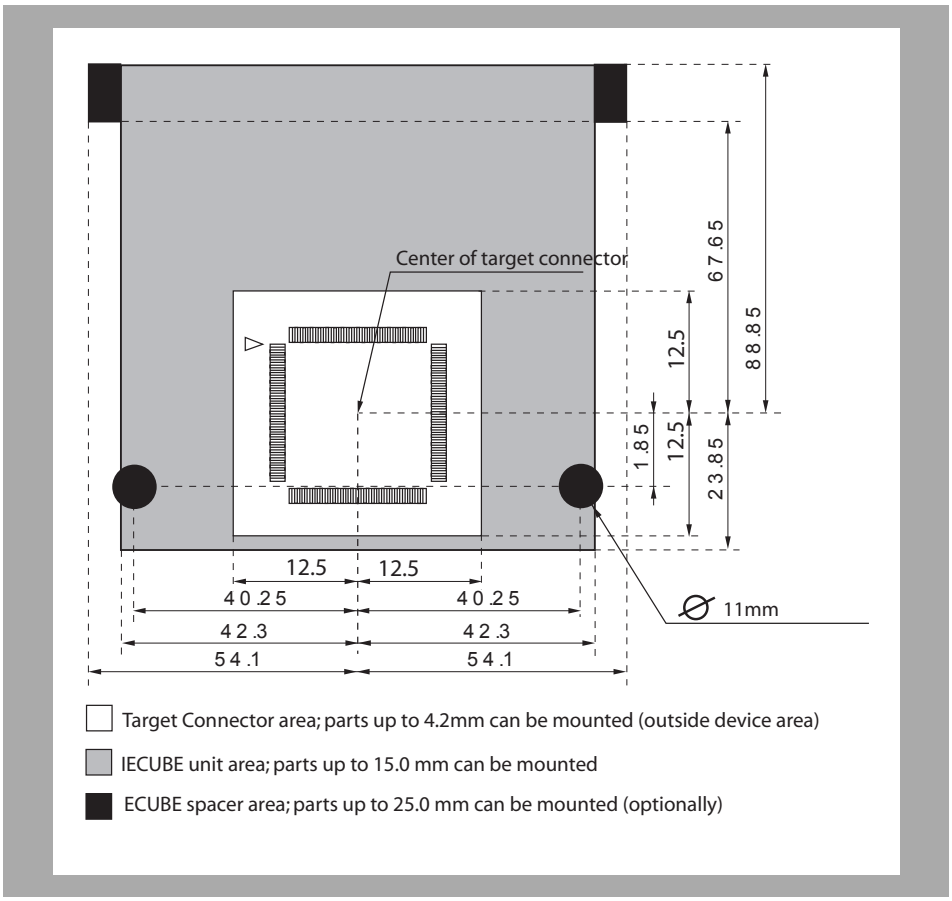


Figure 7-1 Area dimensions - 100-pin

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

7.1.2 Area dimensions - 144-pin

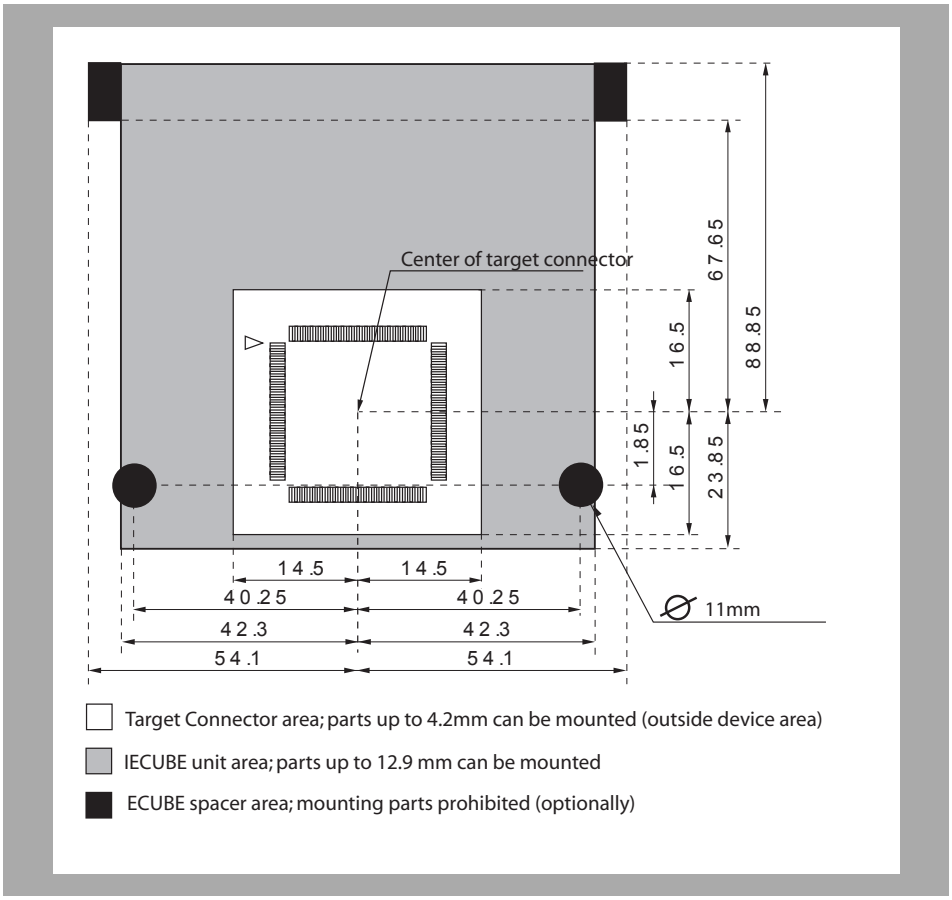


Figure 7-2 Area dimensions - 144-pin

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

7.1.3 Area dimensions -208-pin

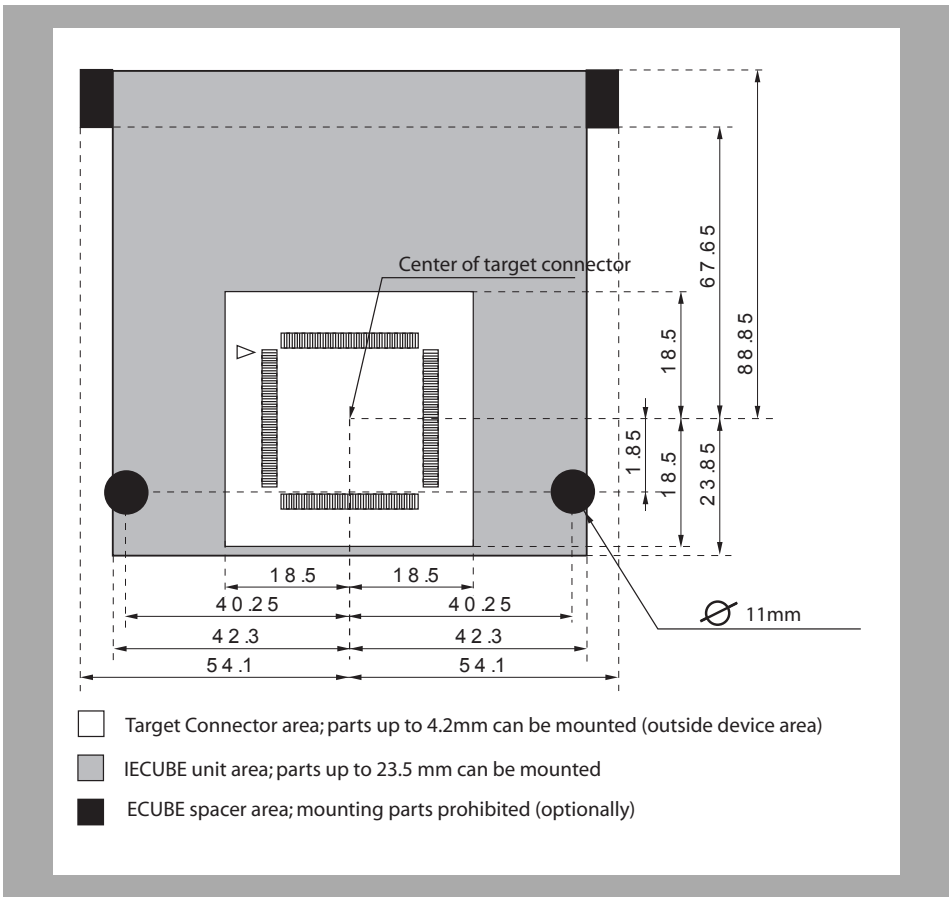


Figure 7-3 Area dimensions - 208-pin

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

7.2 When Extension Probe Is Used

7.2.1 Area dimensions -100-pin with probe

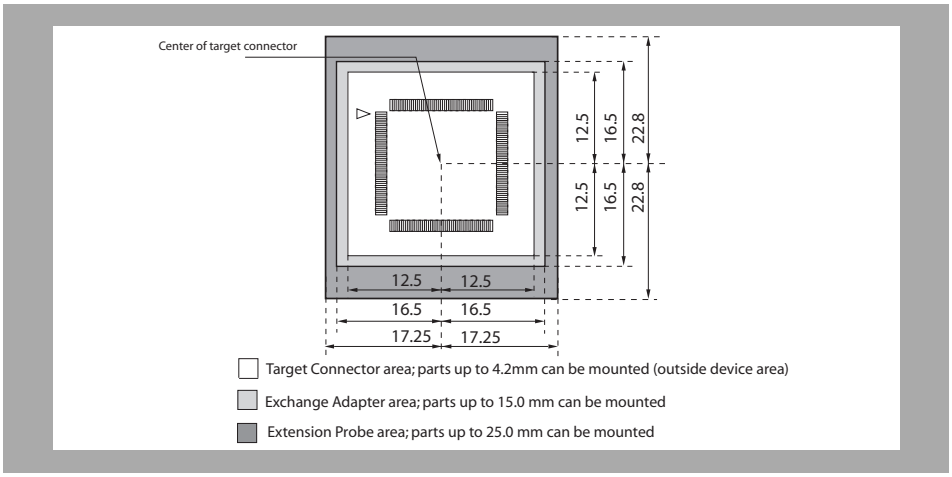


Figure 7-4 Area dimensions - 100-pin with probe

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

7.2.2 Area dimensions -144-pin with probe

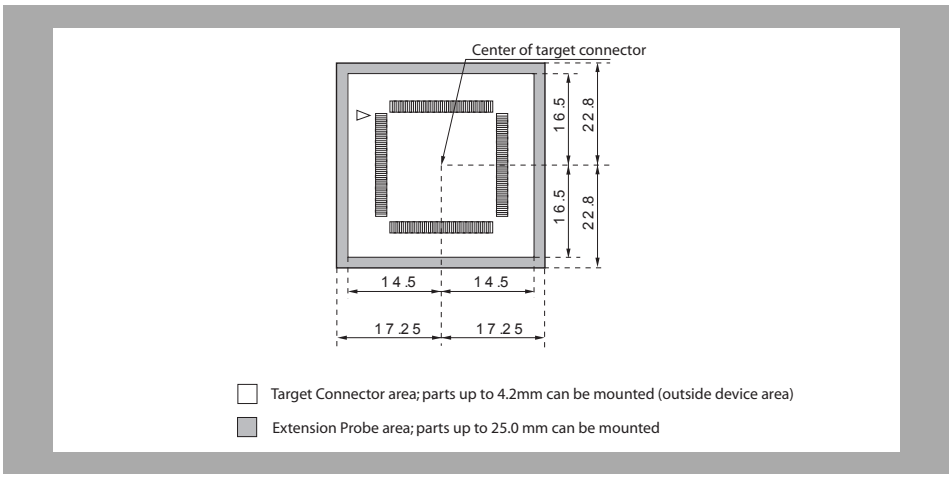


Figure 7-5 Area dimensions - 144-pin with probe

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

7.2.3 Area dimensions -208-pin with probe

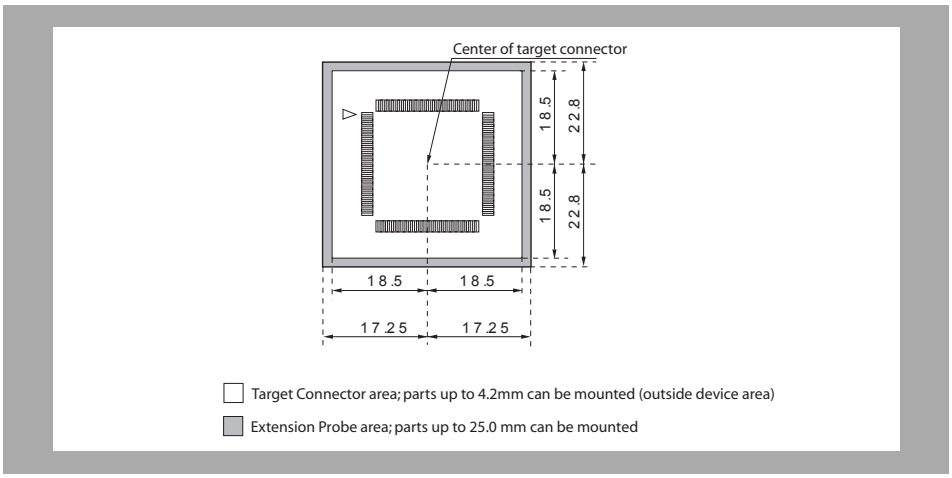


Figure 7-6 Area dimensions - 208-pin with probe

Note The height can be adjusted using the spacer adapter (can increase by 5.6 mm per unit)

Chapter 8 Connector Probe Package Drawings

8.1 Target Connector

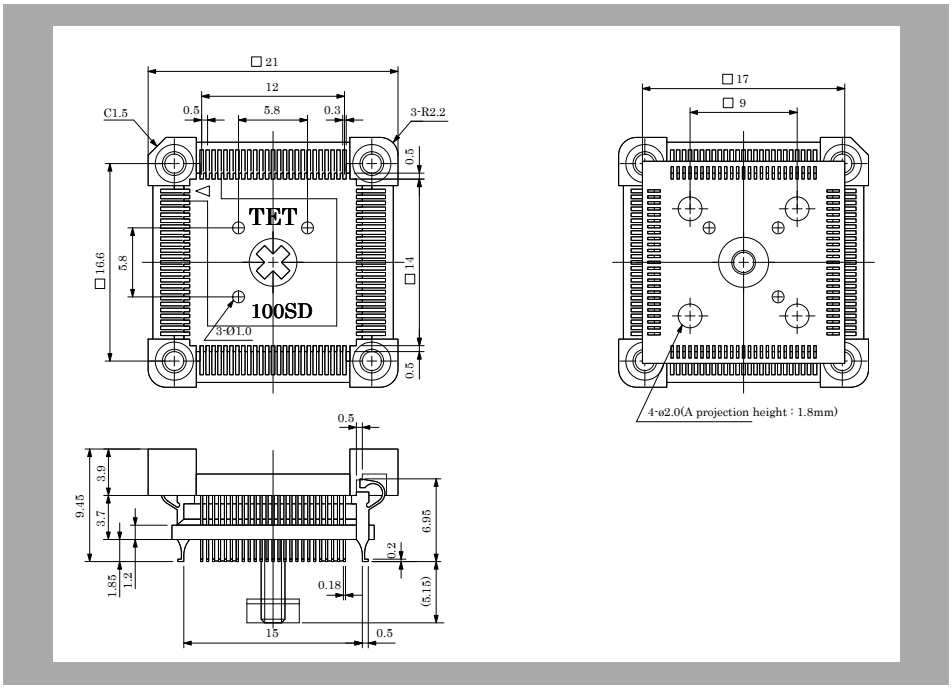


Figure 8-1 Target Connector for 100-pin

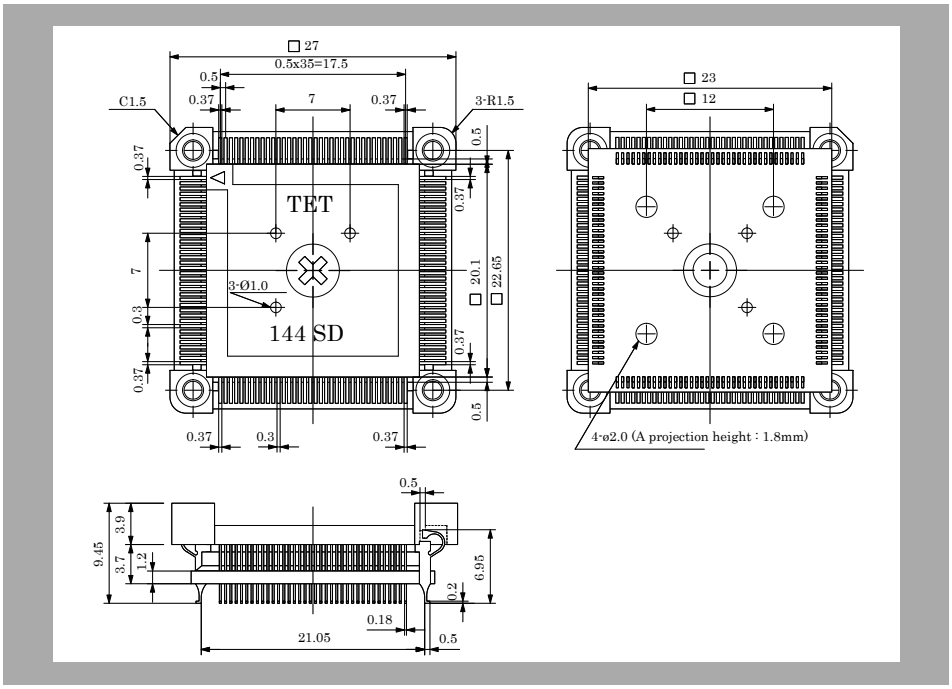


Figure 8-2 Target Connector for 144-pin

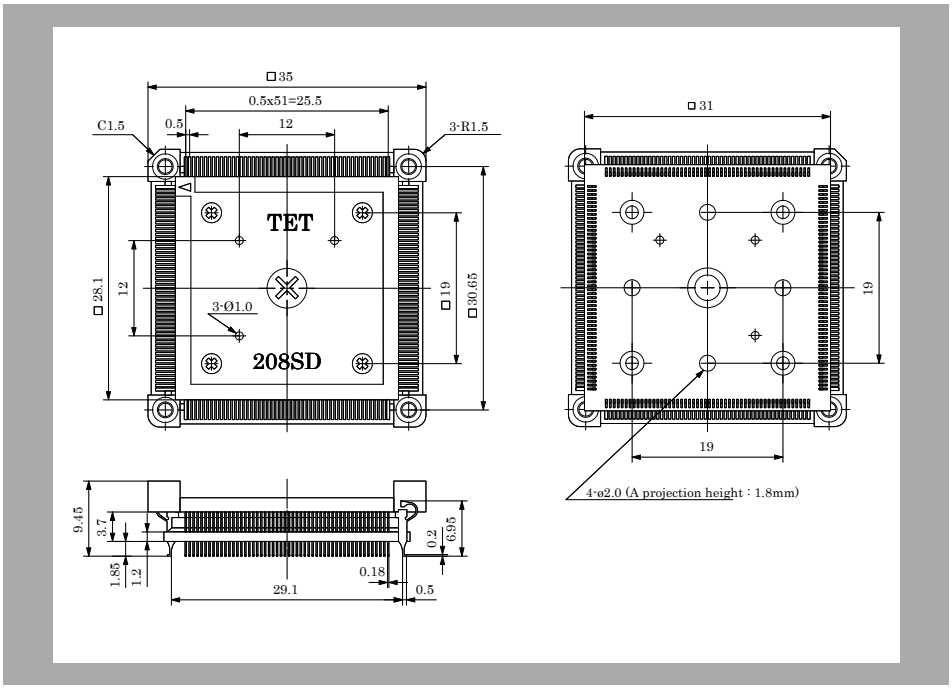


Figure 8-3 Target Connector for 208-pin

8.2 Foot Patterns of Target Connectors

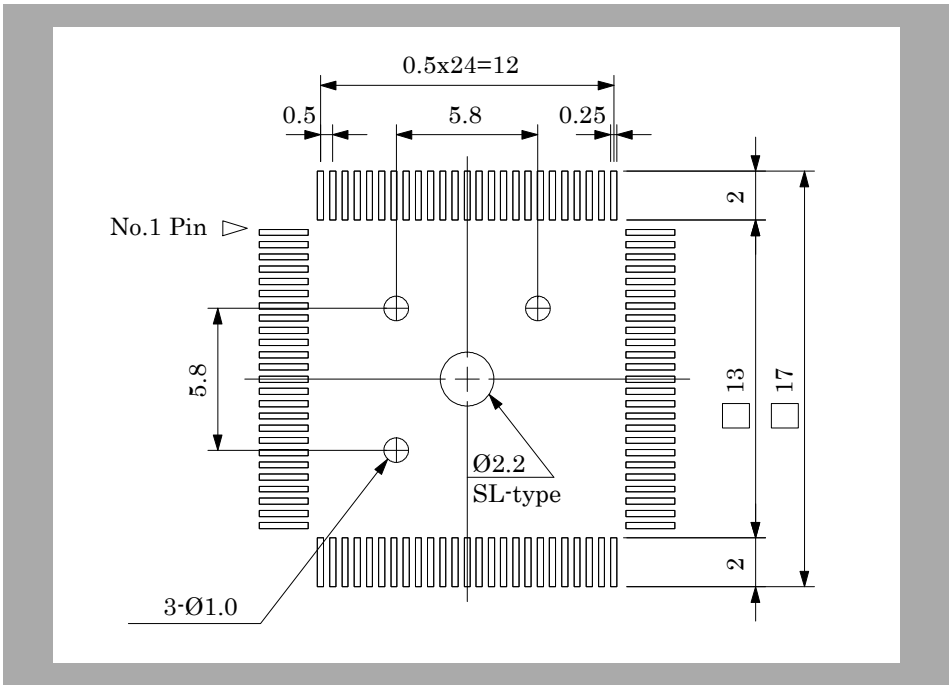


Figure 8-4 Foot Pattern of Target Connector for 100 pin

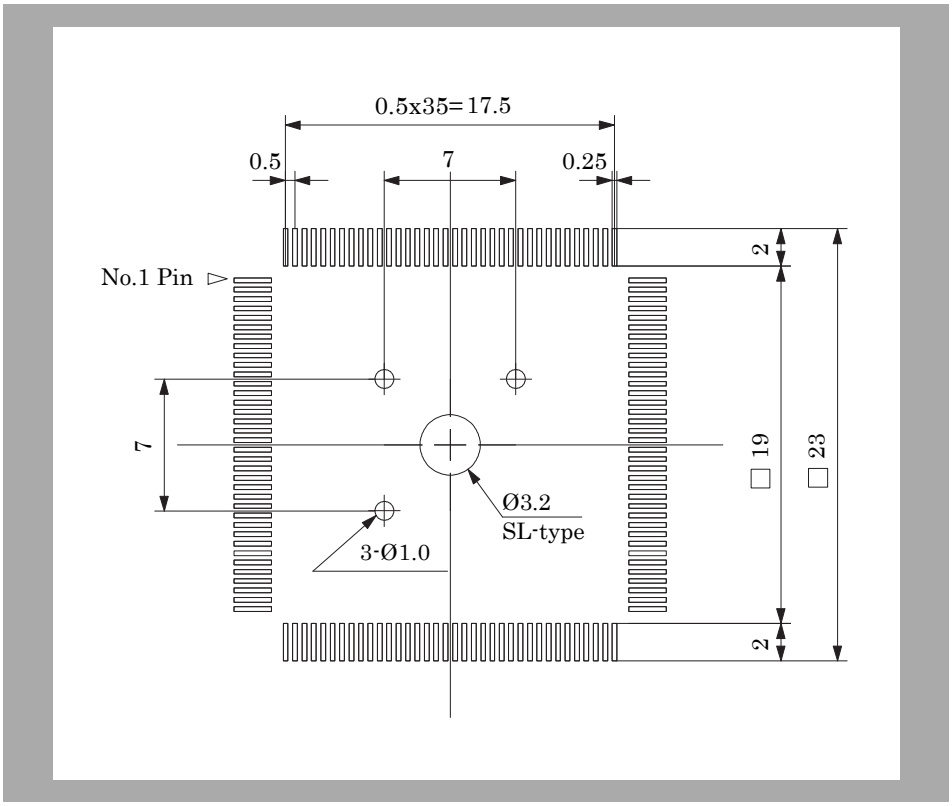


Figure 8-5 Foot Pattern of Target Connector for 144-pin

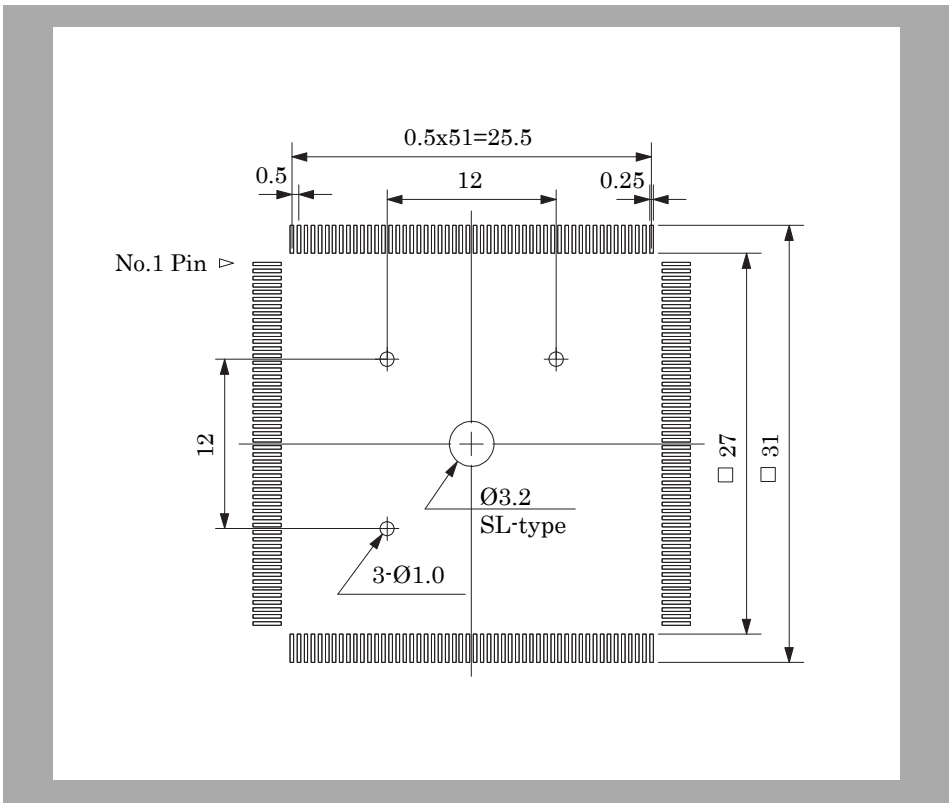


Figure 8-6 Foot Pattern of Target Connector for 208-pin

8.3 Exchange Adapter

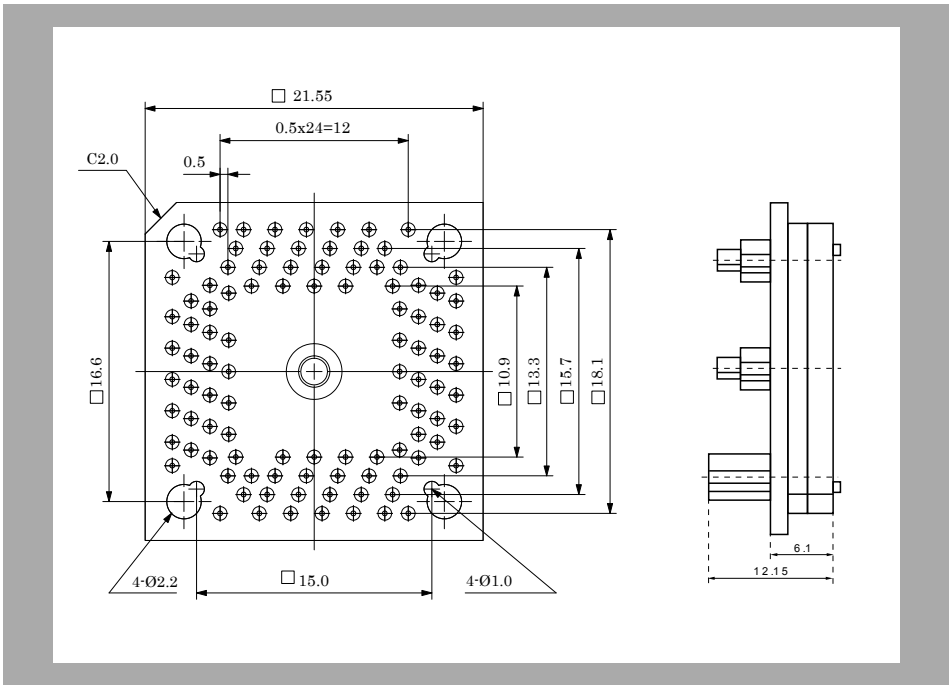


Figure 8-7 Exchange Adapter for 100-pin

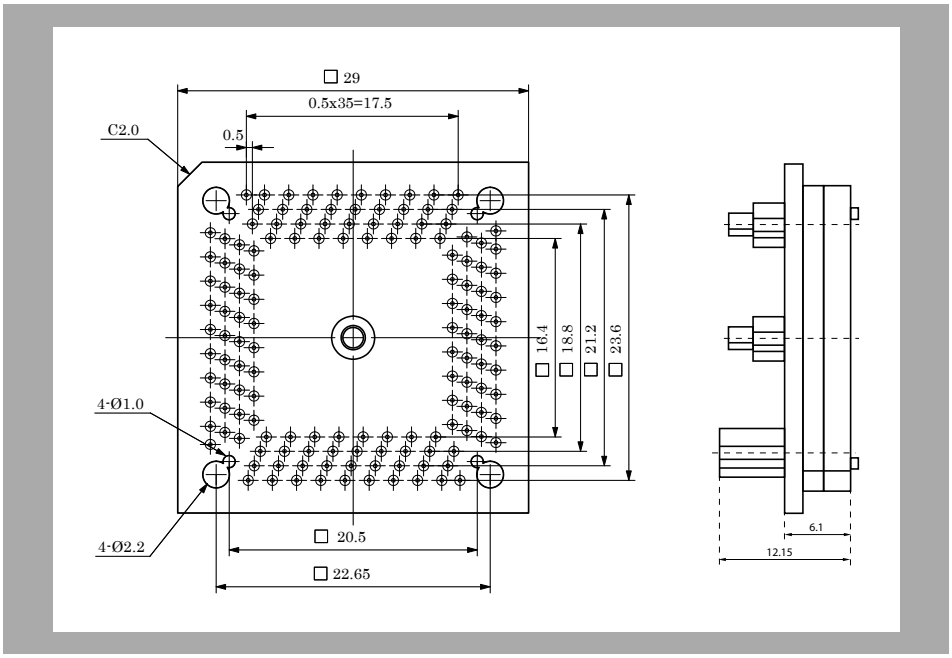


Figure 8-8 Exchange Adapter for 144-pin

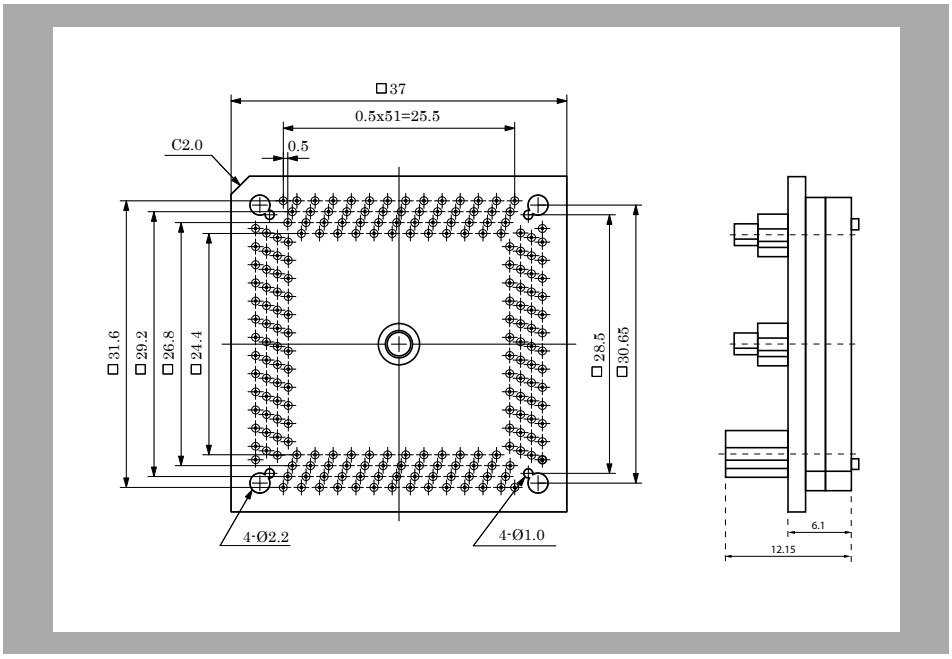


Figure 8-9 Exchange Adapter for 208-pin

8.4 Mounting Adapter

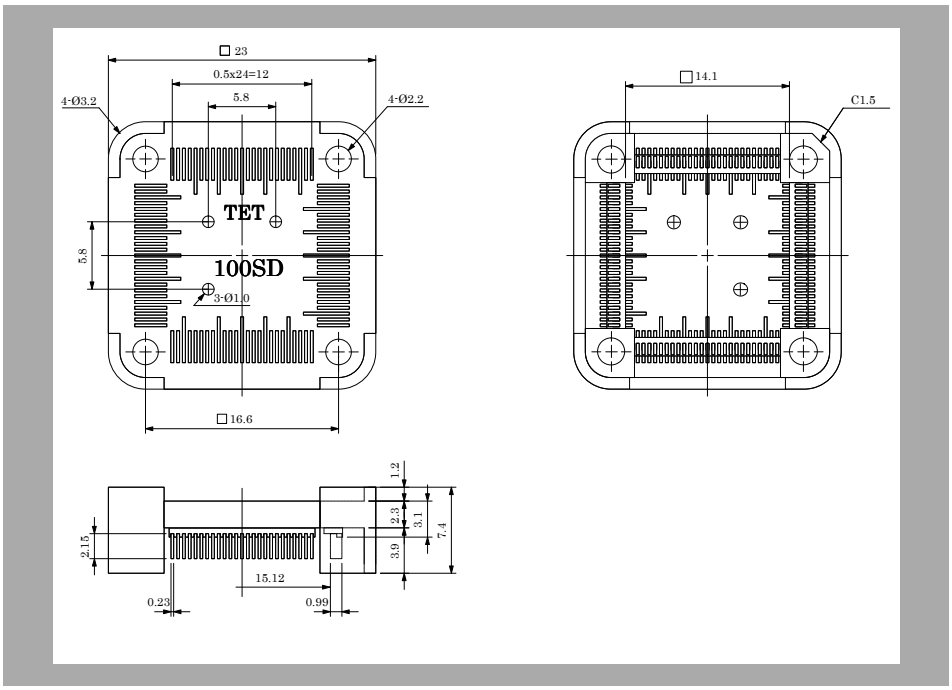


Figure 8-10 Mounting Adapter for 100-pin

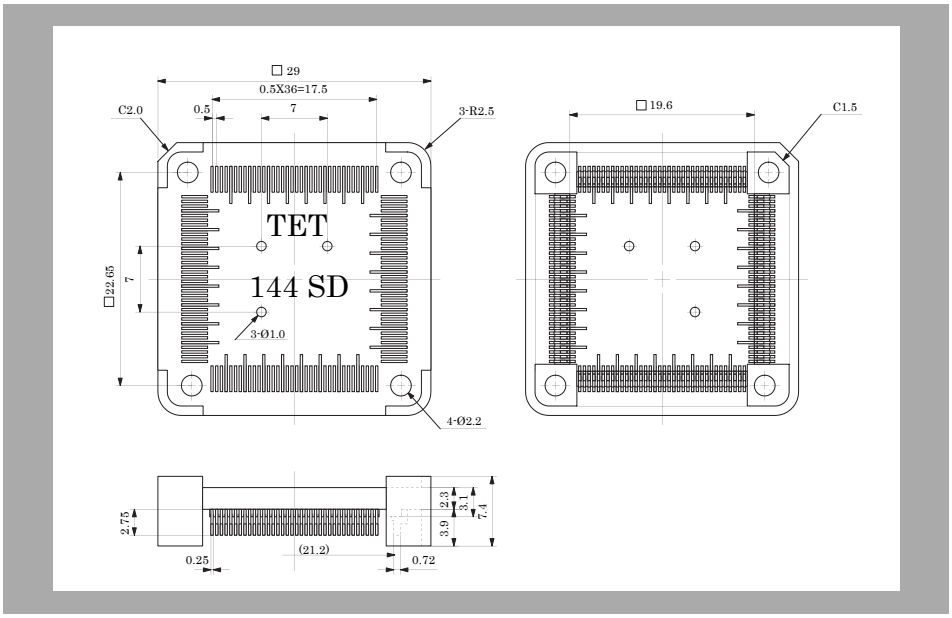


Figure 8-11 Mounting Adapter for 144-pin

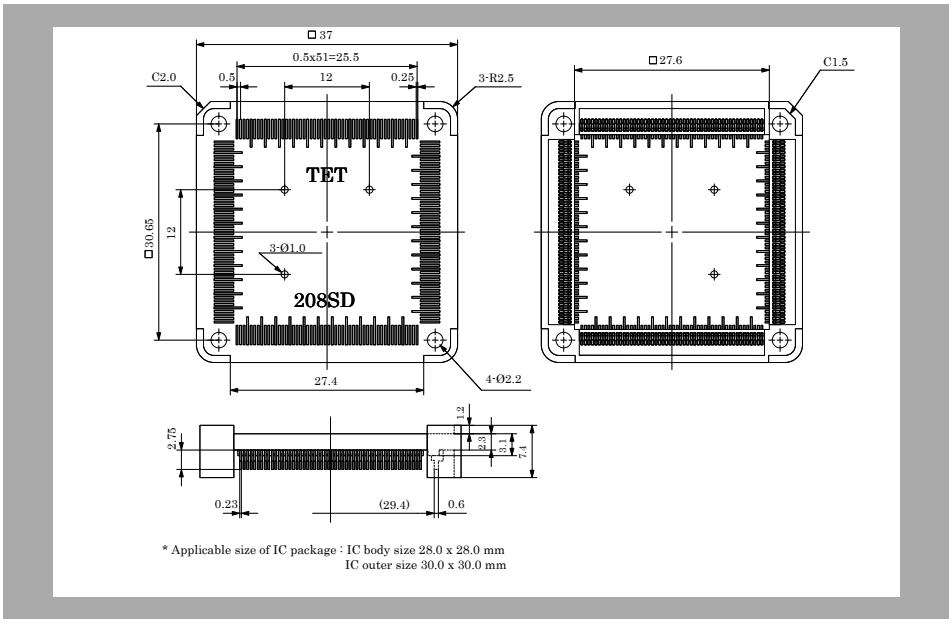


Figure 8-12 Mounting Adapter for 208-pin

8.5 YQ Adapter

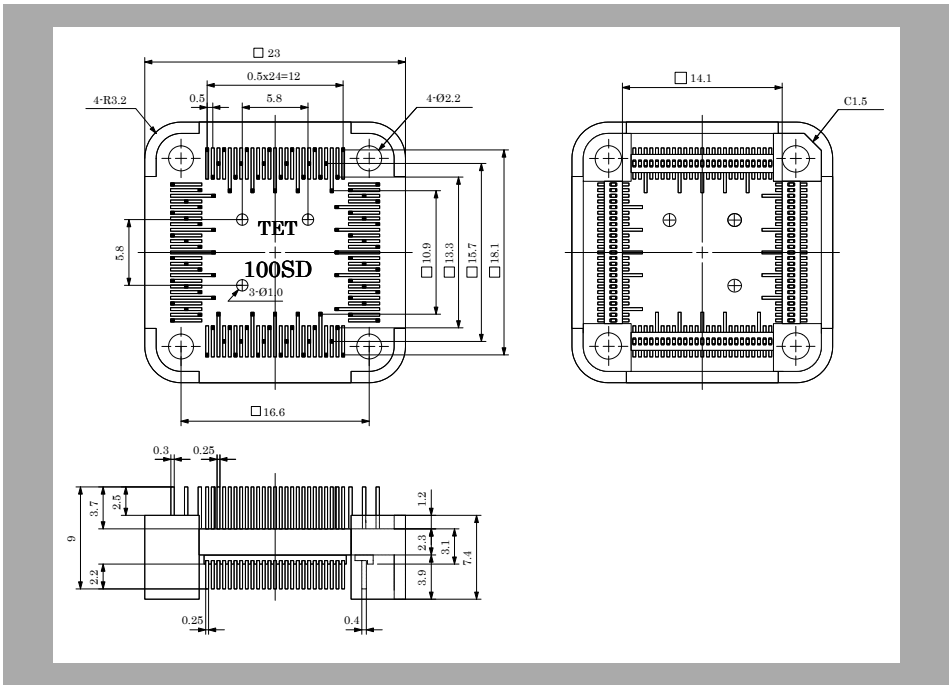


Figure 8-13 YQ Adapter for 100-pin

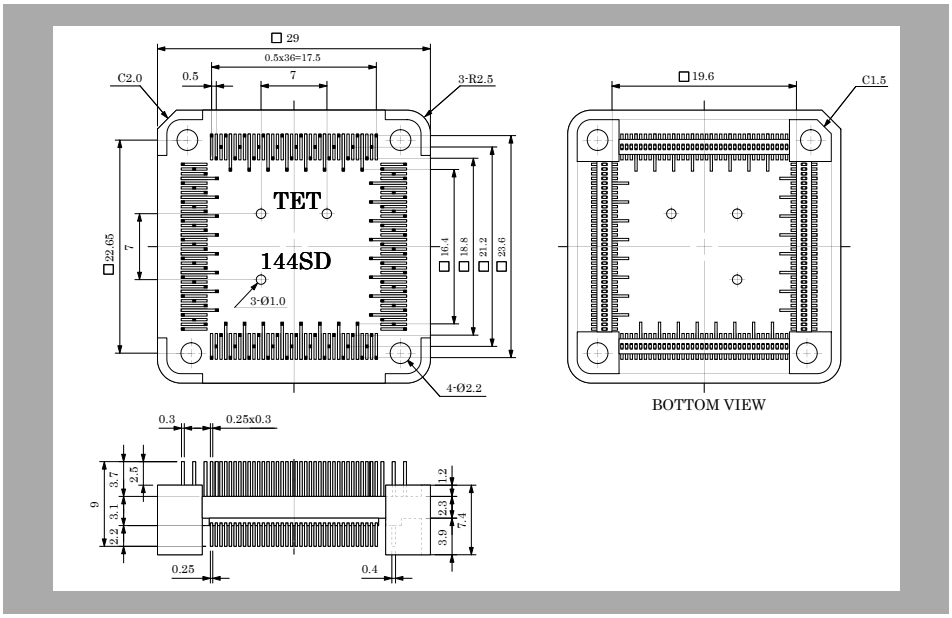


Figure 8-14 YQ Adapter for 144-pin

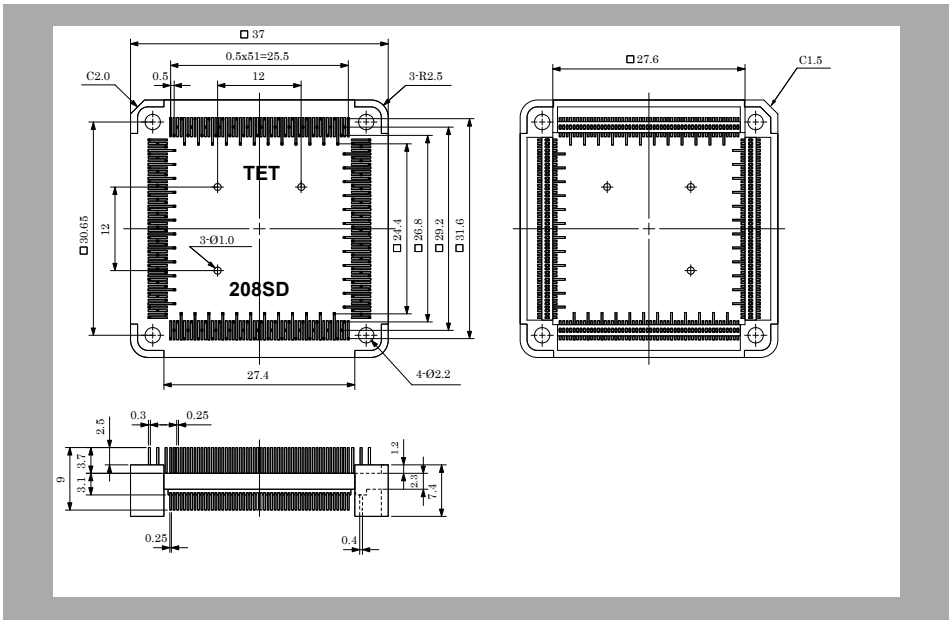


Figure 8-15 YQ Adapter for 208-pin

8.6 Spacer Adapter

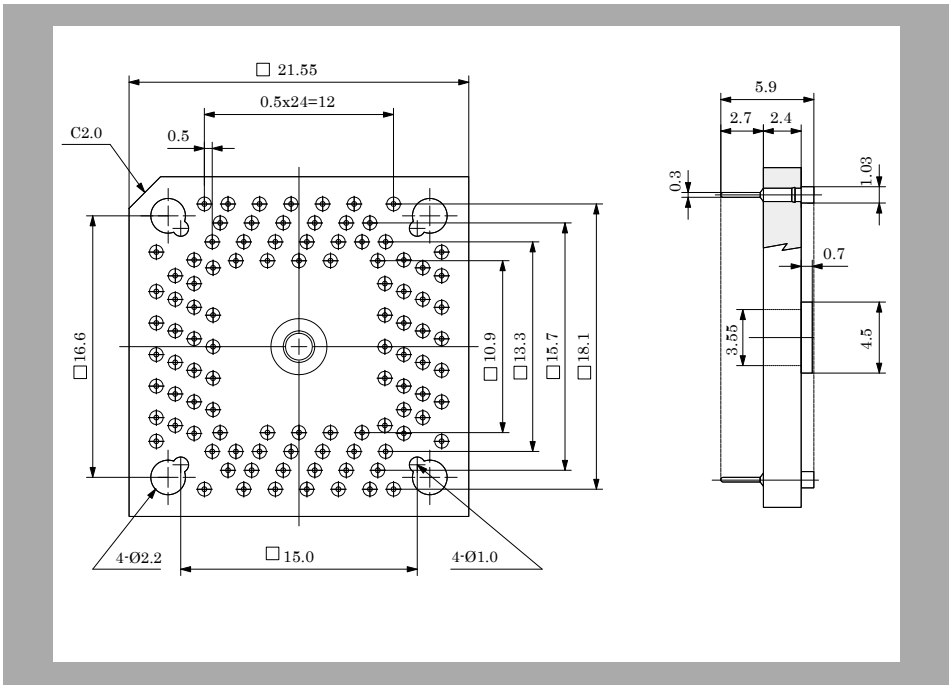


Figure 8-16 Spacer Adapter for 100-pin

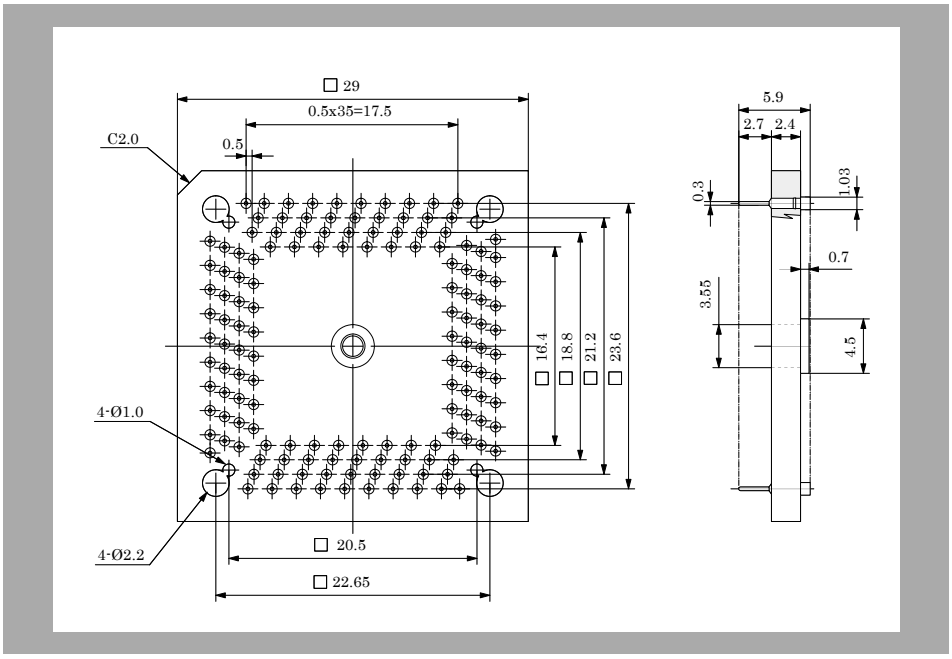


Figure 8-17 Spacer Adapter for 144-pin

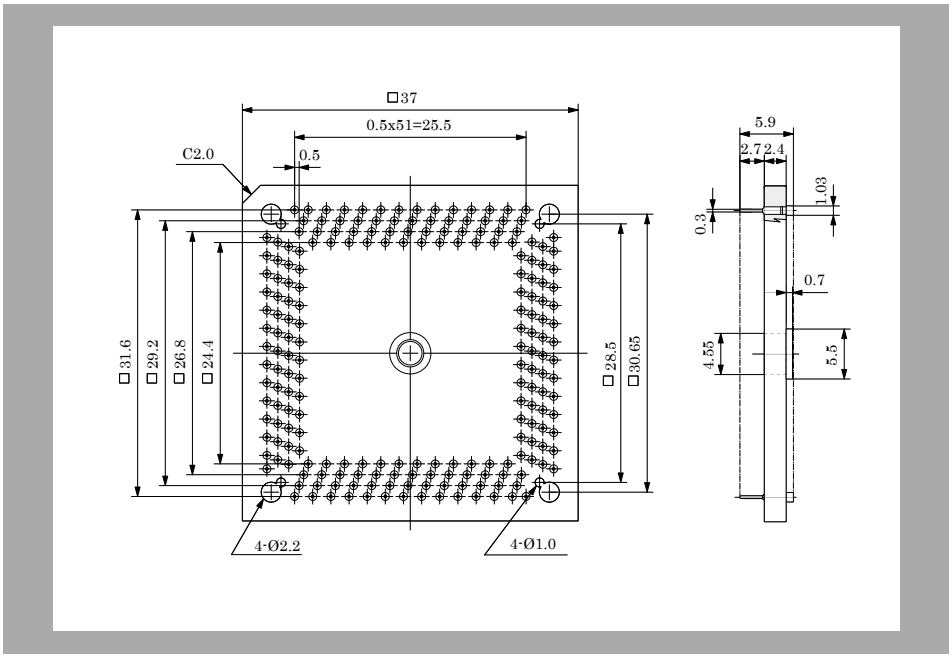


Figure 8-18 Spacer Adapter for 208-pin

8.7 Extension Probe

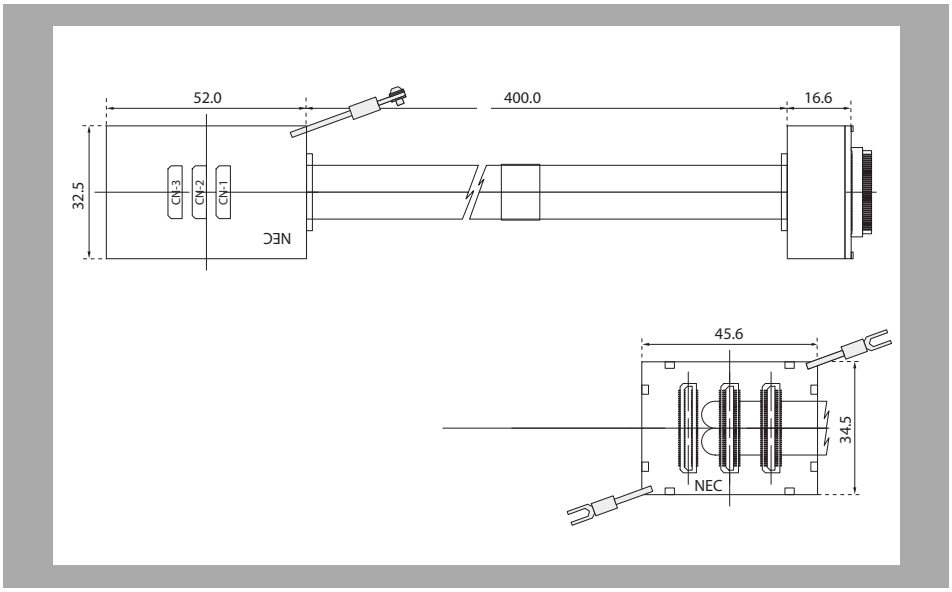


Figure 8-19 Extension Probe

Revision History

This revision list shows all functional changes compared to the previous manual version U18350EE1V0UM00 (date published 04/20/07).

Chapter	Page	Description
2	18	document updated with 100pin device line for μPD70(F)3416 and μPD(F)703417
2	18	chapter system configuration updated with 100pin device line
2	19	all ROM mask devices (μPD703420, μPD703421, μPD703422) and flash device μPD70F3420 removed
3	21	clock name changed to low-speed on-chip oscillator
3	27	chapter software setup updated
5	44	chapter added: Debug Function
6	57	clock name changed to low-speed on-chip oscillator
6	58	document updated with 100pin device line for μPD70(F)341x (x = 6/7)
6	61	clock name changed to low-speed on-chip oscillator
6	66	clock name changed to low-speed on-chip oscillator
6	69	chapter added: Macro's supporting peripheral break function
7	70	drawings updated with 100pin device adapter drawings