

# HI7300/PX

## Large Memory Pool

R20UT0828EJ0100

Rev.1.00

Oct. 14, 2011

### Introduction

This document explains the large memory pool function that is supported by following products.

- HI7300/PX V.1.03 Release 00 or later

Note, the contents of this document are not reflected in the user's manual of the product.

### Contents

1.	Summary .....	2
2.	Service Call.....	3
2.1	Creates Large Memory Pool (vcre_Impl, ivcre_Impl) .....	4
2.2	Deletes Large Memory Pool (vdel_Impl) .....	7
2.3	Acquires Memory Block (vpget_Impl, ivpget_Impl) .....	8
2.4	Releases Memory Block (vrel_Impl, ivrel_Impl) .....	10
2.5	Refers to Large Memory Pool State (vref_Impl, ivref_Impl, vref_Impl2, ivref_Impl2) .....	11
3.	Estimation of Resource Pool Size .....	13
4.	Configuration .....	14

## 1. Summary

The large memory pool function is the implementation to improve the processing time of the variable-size memory pool function.

A big difference with the variable-size memory pool is not to support the function to wait for the acquisition of memory block. Table 1 shows function difference with variable-size memory pool.

**Table 1 Function Difference with Variable-size Memory Pool**

Item		Variable-size Memory Pool	Large Memory Pool
Number of memory pools		Multiple memory pools can be used.	1
Attribute		VTA_UNFRAGMENT can be specified.	None
Service call	Create	cre_mpl, icre_mpl, acre_mpl, iacre_mpl	vcre_impl, ivcre_impl
	Delete	del_mpl	vdel_impl
	Acquire (polling)	pget_mpl, ipget_mpl	vpget_impl, ivpget_impl
	Acquire (wait)	get_mpl	None
	Acquire (wait with time-out)	tget_mpl	None
	Release	rel_mpl, irel_mpl	vrel_impl, ivrel_impl
	Refer	ref_mpl, iref_mpl	vref_impl, ivref_impl, vref_impl2, ivref_impl2

## 2. Service Call

This chapter explains the specification of service calls for large memory pool by the same form as the manual.

**Table 2 Service Calls for large Memory Pool**

Service Call	Description	System State <sup>*1</sup>
		T/N/E/D/U/L/C
vcre_Impl	Creates large memory pool	T/E/D/U
ivcre_Impl		N/E/D/U
vdel_Impl	Deletes large memory pool	T/E/D/U
vpget_Impl	Acquires memory block	T/E/D/U
ivpget_Impl		N/E/D/U
vrel_Impl	Releases memory block	T/E/D/U
ivrel_Impl		N/E/D/U
vref_Impl	Refers to large memory pool state	T/E/D/U
ivref_Impl		N/E/D/U
vref_Impl2	Refers to large memory pool state (Simple version)	T/E/D/U
ivref_Impl2		N/E/D/U

Notes: 1 T: Can be called from task context  
 N: Can be called from non-task context  
 E: Can be called from dispatch-enabled state  
 D: Can be called from dispatch-disabled state  
 U: Can be called from CPU-unlocked state  
 L: Can be called from CPU-locked state  
 C: Can be called from CPU exception handler

## 2.1 Creates Large Memory Pool (vcre\_impl, ivcre\_impl)

### C-Language API:

```
ER ercd = vcre_impl(VT_CLMPL *pk_clmpl);
```

```
ER ercd = ivcre_impl(VT_CLMPL *pk_clmpl);
```

### Parameters:

VT_CLMPL *pk_clmpl	R4	Pointer to the packet where the large memory pool creation information is stored
--------------------	----	----------------------------------------------------------------------------------

### Return Parameters:

ER ercd	R0	Normal end (E_OK) or error code
---------	----	---------------------------------

### Packet Structure:

```
typedef struct {
    ATR    implatr;      +0    4    Large memory pool attribute
    SIZE   implsz;       +4    4    Size of the large memory pool (Number of bytes)
    VP     impl;         +8    4    Start address of the large memory pool area
    UINT   minblksz;     +12   4    Minimum block size
    UINT   sctnum;       +16   4    Maximum number of sectors
} VT_CLMPL;
```

### Error Codes:

E_RSATR	[k]	Reserved attribute
E_PAR	[k]	Parameter error <ul style="list-style-type: none"> <li>(1) pk_clmpl is other than a multiple of four</li> <li>(2) implsz is other than a multiple of four</li> <li>(3) implsz ≥ H'80000000</li> <li>(4) impl is other than a multiple of four if impl is not NULL</li> <li>(5) minblksz is neither 8, 16, 32, 64, 128, 256, 512, 1024, 2048 nor 4096.</li> <li>(6) sctnum == 0</li> <li>(7) implsz &lt; minblksz × 32</li> <li>(8) minblksz == 8 if VTA_ALIGN16 is specified</li> <li>(9) minblksz == 8 or 16 if VTA_ALIGN32 is specified</li> </ul>
E_NOMEM	[k]	Insufficient memory <ul style="list-style-type: none"> <li>(1) Insufficient space in the system pool</li> <li>(2) Insufficient space in the resource pool</li> </ul>
E_OBJ	[k]	Object status is invalid (Large memory pool already exists)
E_NOSPT	[k]	No support <p>Compiler option "-def=USE_LMPL" is not specified for "kernel_def.c"</p>
E_MACV	[k]	Memory access violation

Note : The context error (E\_CTX) is not detected when this service call is called from the system state that is not permitted.

**Function:**

These service calls create the large memory pool.

Note, the large memory pool cannot be created by using the configurator.

**(1) Implatr**

For alignment of the addresses of memory blocks to be acquired from the large memory pool, any one of the following attributes can be specified when necessary.

- VTA\_ALIGN16 (H'00000010): Memory block addresses are adjusted to 16-byte boundaries.
- VTA\_ALIGN32 (H'00000020): Memory block addresses are adjusted to 32-byte boundaries.

When neither one of them is specified, memory block addresses are adjusted to 4-byte boundaries.

**(2) Implsz**

Parameter Implsz specifies the size of the large memory pool to be created.

**(3) Impl**

Parameter Impl specifies the start address of a free area to be used as the large memory pool. The kernel manages Implsz-byte area starting from address Impl as the large memory pool.

When attribute VTA\_ALIGN16 or VTA\_ALIGN32 is specified, the actual large memory pool area to be used starts from an address obtained by adjusting address Impl to a 16-bytes or 32-bytes boundary, which means that the usable memory pool size decreases for the adjusted size.

Note that the kernel does not check which domain can access the specified area. For example, if an address in the P1 or P2 area is specified for the large memory pool area, the area cannot be accessed from a user domain, but the kernel does not detect it.

---

**When the memory object protection function is selected:**

An area that can be read or written to from the kernel domain must be specified as the large memory pool area. If this rule is violated, an E\_MACV error will be returned.

---

When NULL is specified for Impl, the kernel allocates an Implsz-byte large memory pool area in the system pool.

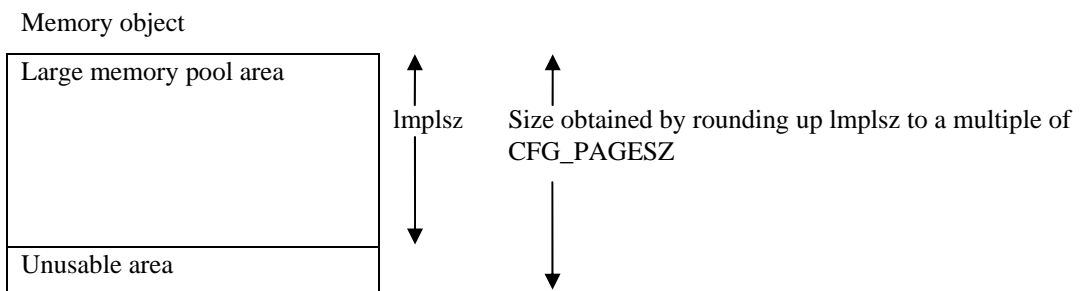
---

**When the memory object protection function is selected:**

The large memory pool area allocated in the system pool by the kernel is a memory object having the following attributes.

- (1) Size: Implsz is rounded up to a multiple of CFG\_PAGESZ.

However, note that only the Implsz bytes can be used for the large memory pool. As shown in the following figure, since the unusable area is included in the memory object where the large memory pool area is allocated, the unusable area has the same access permission as the large memory pool area, but the unusable area is not handled as the large memory pool.



- (2) Domain: When the service call is issued in a task context, the domain of the issuing task is assigned to the large memory pool, which is the same as the domain ID that can be obtained by calling `get_did`. When the service call is issued in a non-task context, the kernel domain is assigned.
- (3) Memory attribute: `TA_RW|TA_CACHE|TA_WBACK`
- (4) Access permission vector: An appropriate vector is specified so that only the assigned domain can read or write to the large memory pool as follows.
  - For the kernel domain: `TACT_KERNEL`
  - For a user domain: `TACT_PRW(domid)`  
(`domid` is the ID of the domain to which the issuing task belongs.)

#### (4) minblksz and sctnum

In the large memory pool, minute memory blocks are continuously arranged, and this is managed as sector. As a result, the fragmentation is reduced.

The size required for `Implmb` grows though minute memory block can be efficiently handled by enlarging `sctnum`.

The size of memory block that can be managed as sector is  $\text{minblksz} \times 8$  (bytes) or less.

When `sctnum` is set to a larger value than  $\text{Implsz} / (\text{minblksz} \times 32)$ ,  $\text{Implsz} / (\text{minblksz} \times 32)$  is assumed.

This service call is a function not defined in the  $\mu$ ITRON4.0 specification.

#### Error Detection through CFG\_MEMCHK:

An `E_MACV` error will be returned in the following cases.

- (1) The domain of the caller does not have a read access permission for `pk_clmpl`, which means that an error will be returned if `prb_mem` is issued with the following parameters.
  - `base = pk_clmpl`
  - `size = sizeof(VT_CLMPL)`
  - `domid = Domain of the caller`
  - `pmmode = TPM_READ`
- (2) When `pk_clmpl->Impl != NULL`, the kernel does not have a read/write access permission for the `Implsz`-byte area starting from address `Impl`, which means that an error will be returned if `prb_mem` is issued with the following parameters.
  - `base = pk_clmpl->Impl`
  - `size = pk_clmpl->Implsz`
  - `domid = Kernel domain`
  - `pmmode = TPM_READ|TPM_WRITE`

## 2.2 Deletes Large Memory Pool (vdel\_impl)

### C-Language API:

```
ER ercd = vdel_impl(void);
```

### Parameters:

None

### Return Parameters:

ER	ercd	R0	Normal end (E_OK) or error code
----	------	----	---------------------------------

### Error Codes:

E_NOEXS	[k]	Undefined (The large memory pool does not exist)
E_CTX	[k]	Context error (Called from the system state that is not permitted)
E_NOSPT	[k]	No support

Compiler option "-def=USE\_LMPL" is not specified for "kernel\_def.c"

### Function:

This service call deletes the large memory pool.

The large memory pool area allocated in the system pool and the management area allocated in the resource pool are released.

The kernel will not perform any processing even when memory blocks have already been acquired.

This service call is a function not defined in the  $\mu$ ITRON4.0 specification.

## 2.3 Acquires Memory Block (vpget\_impl, ivpget\_impl)

### C-Language API:

```
ER ercd = vpget_impl(UINT blksz, VP *p_blk, VP *p_key);
```

```
ER ercd = ivpget_impl(UINT blksz, VP *p_blk, VP *p_key);
```

### Parameters:

UINT	blksz	R4	Memory block size (Number of bytes)
VP	*p_blk	R5	Pointer to the area where the start address of the memory block is to be returned
VP	*p_key	R6	Pointer to the area where the memory block management key is to be returned

### Return Parameters:

ER	ercd	R0	Normal end (E_OK) or error code
----	------	----	---------------------------------

### Error Codes:

E_PAR	[k]	Parameter error (1) p_blk is other than a multiple of four (2) blksz is other than a multiple of four or 0 (3) $\text{implsz}^1 < \text{blksz}$ (4) p_key is other than a multiple of four
E_TMOUT	[k]	Polling Fail (There is no free area with blksz bytes)
E_NOEXS	[k]	Undefined (The large memory pool does not exist)
E_NOSPT	[k]	No support Compiler option "-def=USE_LMPL" is not specified for "kernel_def.c"
E_NOMEM	[k]	Insufficient memory Insufficient space in the resource pool
E_MACV	[k]	Memory access violation

Note : The context error (E\_CTX) is not detected when this service call is called from the system state that is not permitted.

### Function:

These service calls acquire a memory block with the size specified by blksz (number of bytes) from the large memory pool, and returns the start address of the acquired memory block to the area indicated by p\_blk.

The blksz is rounded up as shown in table 3.

<sup>1</sup> Large memory pool size specified by vcre\_impl or ivcre\_impl



**Table 3 Rounding up blksize**

<b>VTA_ALIGN16 attribute</b>	<b>VTA_ALIGN32 attribute</b>	<b>Rounding up blksize</b>
Not specified	Not specified	(1) When $\text{blksize} \leq (\text{minblksize}^2 * 8)$ : The blksize is rounded up to the power of 2 times the value of minblksize. (2) When $\text{blksize} > (\text{minblksize} * 8)$ : The blksize is rounded up to a multiple of 4.
Specified	Not specified	(1) When $\text{blksize} \leq (\text{minblksize} * 8)$ : The blksize is rounded up to the power of 2 times the value of minblksize. (2) When $\text{blksize} > (\text{minblksize} * 8)$ : The blksize is rounded up to a multiple of 16.
Not specified	Specified	(1) When $\text{blksize} \leq (\text{minblksize} * 8)$ : The blksize is rounded up to the power of 2 times the value of minblksize. (2) When $\text{blksize} > (\text{minblksize} * 8)$ : The blksize is rounded up to a multiple of 32.

After the memory block has been acquired, the size of the free space in the large memory pool will decrease by the size of rounded blksize.

For the large memory pool with attribute VTA\_ALIGN16 or VTA\_ALIGN32, the memory block address is a 16-byte or 32-byte boundary address, respectively.

The kernel consumes an area in the resource pool to manage the memory blocks. If there is not sufficient free space in the resource pool, an E\_NOMEM error will be returned immediately.

The memory block management key is returned to the area pointed by p\_key. The memory block management key should be specified at release of the memory block.

This service call is a function not defined in the  $\mu$ TRON4.0 specification.

#### **Error Detection through CFG\_MEMCHK:**

An E\_MACV error will be returned in the following cases.

- (1) The domain of the caller does not have a read/write access permission for p\_blk, which means that an error will be returned if prb\_mem is issued with the following parameters.
  - base = p\_blk
  - size = sizeof(VP)
  - domid = Domain of the caller
  - pmmode = TPM\_READ|TPM\_WRITE
- (2) The domain of the caller does not have a read/write access permission for p\_key, which means that an error will be returned if prb\_mem is issued with the following parameters.
  - base = p\_key
  - size = sizeof(VP)
  - domid = Domain of the caller
  - pmmode = TPM\_READ|TPM\_WRITE

<sup>2</sup> The minblksize indicates the minimum block size specified by vcre\_limpl or ivcre\_limpl.

## 2.4 Releases Memory Block (vrel\_lmpl, ivrel\_lmpl)

### C-Language API:

```
ER ercd = vrel_lmpl(VP blk, VP key);
```

```
ER ercd = ivrel_lmpl(VP blk, VP key);
```

### Parameters:

VP	blk	R4	Start address of memory block
VP	key	R5	memory block management key

### Return Parameters:

ER	ercd	R0	Normal end (E_OK) or error code
----	------	----	---------------------------------

### Error Codes:

E_PAR	[k]	Parameter error (1) blk is other than a multiple of four (2) blk is other than the memory block start address (3) key is illegal.
E_NOEXS	[k]	Undefined (The large memory pool does not exist)
E_NOSPT	[k]	No support Compiler option "-def=USE_LMPL" is not specified for "kernel_def.c"

Note : The context error (E\_CTX) is not detected when this service call is called from the system state that is not permitted.

### Function:

These service calls release a memory block to the large memory pool.

The start address of the memory block acquired by service call `vpget_lmpl` or `ivpget_lmpl` must be specified as parameter `blk`. The memory block management key acquired by service call `vpget_lmpl` or `ivpget_lmpl` must be specified as parameter `key`.

This service call is a function not defined in the  $\mu$ ITRON4.0 specification.

## 2.5 Refers to Large Memory Pool State (vref\_lmpl, ivref\_lmpl, vref\_lmpl2, ivref\_lmpl2)

### C-Language API:

```
ER ercd = vref_lmpl(T_RMPL *pk_rlmpl)
ER ercd = ivref_lmpl(T_RMPL *pk_rlmpl)
ER ercd = vref_lmpl2(T_RMPL *pk_rlmpl)
ER ercd = ivref_lmpl2(T_RMPL *pk_rlmpl)
```

### Parameters:

T_RMPL *pk_rlmpl	R4	Pointer to the packet where the large memory pool state is to be returned
------------------	----	---------------------------------------------------------------------------

### Return Parameters:

ER ercd	R0	Normal end (E_OK) or error code
---------	----	---------------------------------

### Packet Structure:

```
typedef struct {
    ID      wtskid;      +0    4    Wait task ID
    SIZE    fmplsz;      +4    4    Total size of free memory (Number of bytes)
    UINT    fblksz;      +8    4    Maximum free memory size (Number of bytes)
} T_RMPL;
```

### Error Codes:

E_PAR	[k]	Parameter error (1) pk_rlmpl is other than a multiple of four
E_NOEXS	[k]	Undefined (The large memory pool does not exist)
E_NOSPT	[k]	No support Compiler option "-def=USE_LMPL" is not specified for "kernel_def.c"
E_MACV	[k]	Memory access violation

Note : The context error (E\_CTX) is not detected when this service call is called from the system state that is not permitted.

**Function:**

These service calls return the large memory pool state to the area indicated by pk\_rlmpl.

(1) **wtskid**

Always TSK\_NONE(0) is returned.

(2) **fmplsz**

Total size of free memory is returned.

(3) **fblksz**

The free area is usually fragmented. The block up to the size fblksz can be acquired immediately by calling service call vpget\_lmpl or ivpget\_lmpl.

The vref\_lmpl and ivref\_lmpl return the size of the maximum contiguous free area.

The vref\_lmpl2 and ivref\_lmpl2 return the size close to size of the maximum contiguous free area. There is a possibility where a free area that is larger than fblksz.

This service call is a function not defined in the  $\mu$ ITRON4.0 specification.

**Note:**

1. vref\_lmpl, ivref\_lmpl  
The processing time depends on the number of free area.
2. vref\_lmpl2, ivref\_lmpl2  
The worst processing time is fixation.

### 3. Estimation of Resource Pool Size

The following descriptions are added to "13.2.2 When Object is Created" in the "HI7300/PX User's Manual (REJ10J1198-0200)".

#### (8) Large memory pool

##### (a) When allocating the large memory pool from system pool

If the kernel is specified to allocate the pool area when creating the large memory pool, the pool area is allocated from the system pool. However, the resource pool is used for the size of VTSZ\_SPLALCMB at maximum for managing the pool area. The contents of this macro are shown below.

- When CFG\_PROTMEM is selected  

$$\text{VTSZ\_SPLALCMB} = 60$$
- When CFG\_PROTMEM is not selected  

$$\text{VTSZ\_SPLALCMB} = 36$$

##### (b) Large memory pool management

The resource pool is requested for the size of VTSZ\_LMPLMB (Maximum number of sectors). The contents of this macro are shown below

```
VTSZ_LMPLMB(Maximum number of sectors)
= 24 * (Maximum number of sectors) + 224
```

And the following descriptions are added to "13.2.3 Sizes Used and Released at Other Timings".

#### (5) Large memory pool : vpget\_impl, ivpget\_impl

When acquiring a memory block, the resource pool is requested for the size of VTSZ\_BLKMB at maximum. The contents of this macro are shown below.

```
VTSZ_BLKMB = 36
```

This area is released when the block is released or the large memory pool is deleted.

## 4. Configuration

Please do the following to use the large memory pool.

(1) **Compiler option for "kernel\_def.c"**

Specify "-def=USE\_LMPL" for "kernel\_def.c".

## Website and Support

Renesas Electronics Website

<http://www.renesas.com/>

Inquiries

<http://www.renesas.com/inquiry>

All trademarks and registered trademarks are the property of their respective owners.

## Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.  
"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.  
"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.  
"Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.  
(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.  
(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.



### SALES OFFICES

### Renesas Electronics Corporation

<http://www.renesas.com>

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

#### **Renesas Electronics America Inc.**

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

#### **Renesas Electronics Canada Limited**

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

#### **Renesas Electronics Europe Limited**

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K  
Tel: +44-1628-585-100, Fax: +44-1628-585-900

#### **Renesas Electronics Europe GmbH**

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

#### **Renesas Electronics (China) Co., Ltd.**

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

#### **Renesas Electronics (Shanghai) Co., Ltd.**

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

#### **Renesas Electronics Hong Kong Limited**

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

#### **Renesas Electronics Taiwan Co., Ltd.**

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

#### **Renesas Electronics Singapore Pte. Ltd.**

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

#### **Renesas Electronics Malaysia Sdn.Bhd.**

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

#### **Renesas Electronics Korea Co., Ltd.**

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