To all our customers

Regarding the change of names mentioned in the document, such as Mitsubishi Electric and Mitsubishi XX, to Renesas Technology Corp.

The semiconductor operations of Hitachi and Mitsubishi Electric were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Mitsubishi Electric, Mitsubishi Electric Corporation, Mitsubishi Semiconductors, and other Mitsubishi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Note : Mitsubishi Electric will continue the business operations of high frequency & optical devices and power devices.

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

M16C/62

C Compiler Startup Files for the M16C/62 MCU

1. Abstract

The following article describes the Startup files for Mitsubishi's NC30 C compiler. A set of customized Startup files is given for the M30624 version of the M16C/62 microcontroller.

2. Introduction

The Mitsubishi M16C/62 is a 16-bit MCU, based on the M16C CPU core, with an impressive list of features including 10-bit A/D, D/A, UARTS, timers, DMA, and so on, and up to 256k bytes of user flash. The M16C series is ideally suited for programming using the C language.

C compilers for microcontrollers typically require some sort of assembler 'startup' file to set processor modes, initialize variables, and so forth. For the NC30 compiler, the startup file also includes section information so the linker knows where, in physical memory, to put variables, constants, code, and so on. The default files included with the NC30 are "ncrt0.a30", the startup file, and "sect30.inc", which give section information.

3. NCRT0.A30 Description

The NC30 compiler is shipped with a default startup file, "ncrt0.a30". This file is a generic startup, which was written for most of the M16C/60 and M16C/20 series microcontrollers. A customized startup file for the M16C/62 is described in section 7.1.

After reset, execution begins with the code in this startup file. The stack pointer is set to point to a free area in RAM, and the processor mode is set. C requires that all (global) un-initialized variables be set to zero and initialize variables are copied from ROM into RAM.

4. SECT30.INC Description

The NC30 compiler is shipped with a default section definition file, "sect30.inc". This file is a generic section file for the M16C series and typically requires editing for the specific processor. A customized section definition file for the M16C/62 is described in section 7.1.

The purpose of the section definition file is to set the location of the C language sections in the microcontroller's physical memory map. The information here is used by the linker to determine where to put aligned variables (integers), non-aligned variables (characters), code (in ROM), interrupt vectors, and so forth. Figure 1 is an example of a memory map for an M16C/62 program that used the customized startup files. Note that the example program assumes the processor is operating in expanded mode with external RAM at address 10000h and external ROM at 6000h.

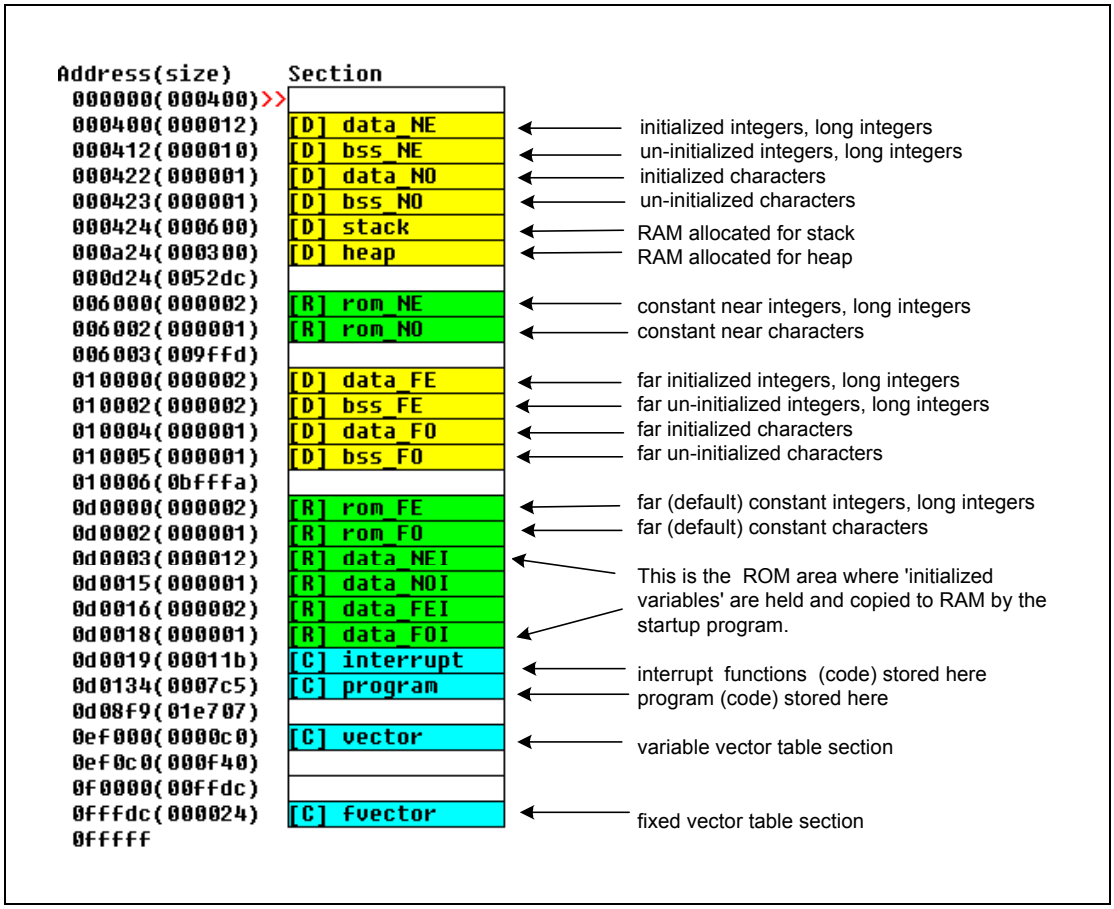

Figure 1. M16C/62 Memory Map of Startup Files

5. Automatic Installation

When starting a new project using “TOOL MANAGER” (Mitsubishi’s development environment), the project wizard will ask if you wish to have the default startup files copied into the project’s working directory. In order to have the project wizard copy the custom files instead, replace the default files with the custom startup files in the directory:

c:\MTOOL\SRC30\STARTUP

This assumes that when you installed the compiler, the default directory c:\MTOOL was specified. It is strongly recommended that you back up the default files first. Also, if you installed the development tools from a “Starter Kit” CD, the custom startup files included with the kit will automatically be used.

6. References

- data sheets: 62aeds.pdf
- C Language Programming Manual: 6020EC.PDF

7. Software Code

7.1 Customized Startup Files for the M16C/62

The following is a set of customized startup files for the M30624 MCU. Except for adding entries into the interrupt vector tables, these files should suffice as-is for most applications. If using different versions of the M16C/62, the ROM starting address will need to be modified.

```
;======================================================================
;             NC30 C COMPILER for M16C/60 and M16C/20 series MCU's
;
;             Name:  ncrt0.a30
;      description: Customized startup program for the M16C/62 (M30624)
;                   microcontroller using the NC30 compiler. Programs
;                   complied with this ;startup file will run under the
;                   MSV1632 ROM Monitor or 'stand alone'.
;
;
;      Copyright 1995-2001 MITSUBISHI ELECTRIC CORPORATION
;      AND MITSUBISHI ELECTRIC SEMICONDUCTOR SYSTEMS CORPORATION AND
;       MITSUBISHI ELECTRIC AND ELECTRONICS USA
;      All Rights Reserved
;
;
;
;       $Id:
;
;======================================================================

;----------------------------------------------------------------------
; Section allocation and definitions
;----------------------------------------------------------------------

       .list OFF
       .include sect30.inc
       .list ON
```

```
;======================================================================
; Interrupt section start
;----------------------------------------------------------------------
       .insf  start,S,0   ; for stkviewer (see Tool Manager and NC30 manuals)
       .glb   start
       .section     interrupt

protect      .equ   0ah
cm0          .equ   06h
cm1          .equ  07h
pm1          .equ  05h

;----------------------------------------------------------------------
; after reset, execution starts here
;----------------------------------------------------------------------

;Upon reset, the processor clock (bclk) is divided by 8 (f/8). The ROM Monitor
;on the Starter Kit sets bclk to f/1. For consistent stand alone operation,
;bclk is set to f/1 here.

start:
       ldc    #istack_top,isp   ;set istack pointer
       mov.b  #03h,protect      ;need to set protect register to operate on clock
       mov.b  #08h,cm0          ;mode and processor mode registers.
       mov.b   #08h,pm1         ;ROM Monitor sets this bit,set here for stand alone
                                ;operation (allows use of all internal RAM & ROM)
       mov.b  #00h,protect
       ldc        #0000h,    flg   ;ensure using register block 0 and use ISP if no RTOS
       ldc        #stack_top,sp    ;set if using an RTOS, has no effect otherwise
       ldc        #data_SE_top,sb  ;set sb register, for sb relative addressing
       ldintb    #VECTOR_ADR
       nop                              ;Delay before
       fset       i                     ;enabling interrupts.

;======================================================================
; Variable area initialization. This code uses macro's (see sect30.inc)
; for initializing C variables. Clears global variables,
; sets initialized variables, etc.
;----------------------------------------------------------------------
; bss zero clear
;----------------------------------------------------------------------
       N_BZERO      bss_SE_top,bss_SE
       N_BZERO      bss_SO_top,bss_SO
       N_BZERO      bss_NE_top,bss_NE
       N_BZERO      bss_NO_top,bss_NO
```

```
;-----------------------------------------------------------------------
; initialize data section
;-----------------------------------------------------------------------
      N_BCOPY      data_SEI_top,data_SE_top,data_SE
      N_BCOPY      data_SOI_top,data_SO_top,data_SO
      N_BCOPY      data_NEI_top,data_NE_top,data_NE
      N_BCOPY      data_NOI_top,data_NO_top,data_NO

;=======================================================================
; FAR area initialize.
;-----------------------------------------------------------------------
; bss zero clear
;-----------------------------------------------------------------------
      BZERO  bss_FE_top,bss_FE
      BZERO  bss_FO_top,bss_FO

;-----------------------------------------------------------------------
; Copy edata_E(O) section from edata_EI(OI) section
;-----------------------------------------------------------------------
      BCOPY  data_FEI_top,data_FE_top,data_FE
      BCOPY  data_FOI_top,data_FO_top,data_FO

;=======================================================================
; heap area initialize. Can be removed if not using memory allocate
; functions
;-----------------------------------------------------------------------
      .glb   __mbase
      .glb   __mnext
      .glb   __msize
      mov.w  #(heap_top&0FFFFH), __mbase
      mov.w  #(heap_top>>16), __mbase+2
      mov.w  #(heap_top&0FFFFH), __mnext
      mov.w  #(heap_top>>16), __mnext+2
      mov.w  #(HEAPSIZE&0FFFFH), __msize
      mov.w  #(HEAPSIZE>>16), __msize+2

;=======================================================================
; Initialize standard I/O
;-----------------------------------------------------------------------
      .glb   _init
      ;jsr.a _init    ;required if using I/O stream serial port driver

;=======================================================================
; Call main() function
;-----------------------------------------------------------------------
      ldc    #0h,fb ; for debugger on starter kit

      .glb   _main
      jsr.a  _main
```

```
;=====================================================================
; exit() function. This function is used in case of accidental return
; from main() or debugging code could be placed here.
;---------------------------------------------------------------------
       .glb   _exit
       .glb   $exit
_exit:                 ; End program
$exit:
       jmp    _exit

;=====================================================================
; dummy interrupt function. Used for all unassigned interrupts(see end
; of sect30.inc.
;---------------------------------------------------------------------
dummy_int:
       reit

       .end


;***********************************************************************
;
;  sect30.inc :    Customized section and macro definitions for the M30624
;                  (M16C/62) microcontroller using the NC30 compiler.
;
;  Description :  This file is specific to the M30624 microcontroller and adapted
;                 for use with the MSV1632 Starter Kit. UART1 interrupt
;                 vectors are used for the Starter Kit debugger.
;
;
;                   Copyright 1995-2001 MITSUBISHI ELECTRIC CORPORATION
;                   MITSUBISHI ELECTRIC SEMICONDUCTOR SYSTEMS CORPORATION AND
;                   MITSUBISHI ELECTRIC AND ELECTRONICS USA
;                   All Rights Reserved.
;
;
;
;       $Id:
;
;***********************************************************************

;---------------------------------------------------------------------
; HEEP SIZE definition. Only used for memory allocate functions
;  (malloc, realloc, etc). If not required and need this RAM for other
;  usage, reduce the value of HEAPSIZE.
;---------------------------------------------------------------------
HEAPSIZE     .equ   300h
```

```
;-----------------------------------------------------------------------
; STACK SIZE definition. Unless the system is running an RTOS, both
; interrupts and function calls should use the istack only (default startup
; configuration). If not required and need this RAM for other
; usage, reduce the value of USTACKSIZE.
;-----------------------------------------------------------------------
STACKSIZE    .equ   300h

;-----------------------------------------------------------------------
; INTERRUPT STACK SIZE definition
;-----------------------------------------------------------------------
ISTACKSIZE   .equ   300h

;-----------------------------------------------------------------------
; INTERRUPT VECTOR ADDRESS. do not set within a flash memory block used by
; the ROM Monitor.
;-----------------------------------------------------------------------
VECTOR_ADR   .equ   0ef000h

;=======================================================================
; Initialize Macro declarations. These macro's are used in the startup
; file (ncrto.a30)for initializing C variables. Clears global variables,
; sets intialized variables, etc.
;-----------------------------------------------------------------------
N_BZERO      .macro TOP_ ,SECT_
       mov.b  #00H, R0L
       mov.w  #(TOP_ & 0FFFFH), A1
       mov.w  #sizeof SECT_ , R3
       sstr.b
       .endm

N_BCOPY .macro      FROM_,TO_,SECT_
       mov.w  #(FROM_ & 0FFFFH),A0
       mov.b  #(FROM_ >>16),R1H
       mov.w  #TO_ ,A1
       mov.w  #sizeof SECT_ , R3
       smovf.b
       .endm

BZERO  .macro TOP_,SECT_
       push.w #sizeof SECT_ >> 16
       push.w #sizeof SECT_  & 0ffffh
       pusha  TOP_ >>16
       pusha  TOP_ & 0ffffh

       .glb   _bzero
       jsr.a  _bzero
       .endm
```

```
BCOPY  .macro FROM_ ,TO_ ,SECT_
       push.w #sizeof SECT_  >> 16
       push.w #sizeof SECT_  & 0ffffh
       pusha  TO_ >>16
       pusha  TO_  & 0ffffh
       pusha  FROM_ >>16
       pusha  FROM_  & 0ffffh

       .glb   _bcopy
       jsr.a  _bcopy
       .endm

;------------------------------------------------------------------
; Special page definition. For defining routines or functions as
; special page.
;------------------------------------------------------------------
;macro define for special page
;
;Format:
;      SPECIAL      number
;

SPECIAL       .macro NUM
       .org   0FFFFEH-(NUM*2)
       .glb   __SPECIAL_@NUM
       .word  __SPECIAL_@NUM  & 0FFFFH
.endm


;------------------------------------------------------------------
;   Section allocation. The following declarations sets the location of the
;   sections in the physical memory map. DO not change these settings
;   without refering to the NC30 manual on startup files.
;
;------------------------------------------------------------------
; Near RAM data area
;------------------------------------------------------------------
; SBDATA area
       .section     data_SE,DATA
       .org   400H
data_SE_top:
       .glb   __SB__
__SB__:                     ; declare sb 'section' here
       .section     bss_SE,DATA,ALIGN
bss_SE_top:

       .section     data_SO,DATA
```

```
data_SO_top:

       .section     bss_SO,DATA
bss_SO_top:

; near RAM area
       .section     data_NE,DATA,ALIGN
data_NE_top:

       .section     bss_NE,DATA,ALIGN
bss_NE_top:

       .section     data_NO,DATA
data_NO_top:

       .section     bss_NO,DATA
bss_NO_top:


;------------------------------------------------------------
; Stack area. If the USP is not required, and the RAM
; allocated to the USP is needed, do not modify the declarations
; below, Simply set the USTACKSIZE (above) to zero.
;------------------------------------------------------------
       .section     stack,DATA
       .blkb  STACKSIZE
stack_top:

       .blkb  ISTACKSIZE
istack_top:


;------------------------------------------------------------
;   Heap section. If the heap is not required, and the RAM
;   allocated to the heap is needed, do not modify the declarations
;   below, Simply set the HEAPSIZE (above) to zero.
;---------------------.--------------------------------------
       .section     heap,DATA
heap_top:
       .blkb  HEAPSIZE

;------------------------------------------------------------
; Near ROM data area. For "near const".
; By definition, Near ROM is all ROM below address 10000h
;------------------------------------------------------------
;      .org         06000H ; Example. External ROM located at 6000h
       .section     rom_NE,ROMDATA  ;rom_NE,ROMDATA,ALIGN

       .org         06000H ; Example. External ROM located at 6000h
```

```
rom_NE_top:

       .section     rom_NO,ROMDATA
rom_NO_top:

;------------------------------------------------------------------
; Far RAM data area. For "far" int's char's, etc
; By definition, Far RAM is all RAM above address FFFFh
;------------------------------------------------------------------
       .section     data_FE,DATA
       .org         10000H ; Example. External RAM located at 10000h
data_FE_top:

       .section     bss_FE,DATA,ALIGN
bss_FE_top:

       .section     data_FO,DATA
data_FO_top:

       .section     bss_FO,DATA
bss_FO_top:


;------------------------------------------------------------------
; Far ROM data area
;------------------------------------------------------------------
       .section     rom_FE,ROMDATA
;Out of reset, the C0000h flash block (block6) is not visible until
;the pm13 bit is set(see M30624 spec's, Processor Mode Register 1)
;The ROM Monitor sets this bit, but for consistent stand alone
;operation, do not allow the reset vector to point to an address
;below D0000h.
       .org         0d0000H
rom_FE_top:

       .section     rom_FO,ROMDATA
rom_FO_top:

;------------------------------------------------------------------
; Initial data of 'data' section
;------------------------------------------------------------------
       .section     data_SEI,ROMDATA
data_SEI_top:

       .section     data_SOI,ROMDATA
data_SOI_top:

       .section     data_NEI,ROMDATA
```

```
data_NEI_top:

       .section     data_NOI,ROMDATA
data_NOI_top:

       .section     data_FEI,ROMDATA
data_FEI_top:

       .section     data_FOI,ROMDATA
data_FOI_top:


;------------------------------------------------------------------
; Switch Table Section
;------------------------------------------------------------------
       .section        switch_table,ROMDATA
switch_table_top:


;------------------------------------------------------------------
; code area
;------------------------------------------------------------------
       .section     interrupt

       .section     program

       .section     program_S     ; special page code must be in the
       .org         0f0000h       ; address range of F0000h to FFFDCh

;------------------------------------------------------------------
; variable vector section
; For proper interrupt operation, replace "dummy_int" with the assembler
; label or absolute address of the interrupt service routine
;------------------------------------------------------------------
       .section     vector       ; variable vector table
       .org   VECTOR_ADR


       .lword dummy_int           ; BRK  (vector 0)
       .org   (VECTOR_ADR+16)
       .lword dummy_int           ; int3(for user)(vector 4)
       .lword dummy_int           ; timerB5(for user)(vector 5)
       .lword dummy_int           ; timerB4(for user)(vector 6)
       .lword dummy_int           ; timerB3(for user)(vector 7)
       .lword dummy_int           ; si/o4 /int5(for user)(vector 8)
       .lword dummy_int           ; si/o3 /int4(for user)(vector 9)
       .lword dummy_int           ; Bus collision detection(for user)(v10)
       .lword dummy_int           ; DMA0(for user)(vector 11)
       .lword dummy_int           ; DMA1(for user)(vector 12)
```

```
      .lword dummy_int          ; Key input interrupt(for user)(vect 14)
      .lword dummy_int          ; A-D(for user)(vector 14)
      .lword dummy_int          ; uart2 transmit(for user)(vector 15)
      .lword dummy_int          ; uart2 receive(for user)(vector 16)
      .lword dummy_int          ; uart0 transmit(for user)(vector 17)
      .lword dummy_int          ; uart0 receive(for user)(vector 18)
      .lword 0fcb6bh            ; uart1 transmit-used by ROM Monitor(vector 19)
      .lword 0fcb6bh            ; uart1 receive-used by ROM Monitor(vector 20)

      .lword dummy_int          ; timer A0(for user)(vector 21)
      .lword dummy_int          ; timer A1(for user)(vector 22)
      .lword dummy_int          ; timer A2(for user)(vector 23)
      .lword dummy_int          ; timer A3(for user)(vector 24)
      .lword dummy_int          ; timer A4(for user)(vector 25)

      .lword dummy_int          ; timer B0(for user)(vector 26)
      .lword dummy_int          ; timer B1(for user)(vector 27)
      .lword dummy_int          ; timer B2(for user)(vector 28)
      .lword dummy_int          ; int0 (for user)(vector 29)
      .lword dummy_int          ; int1 (for user)(vector 30)
      .lword dummy_int          ; int2 (for user)(vector 31)

      .lword dummy_int          ; vector 32 (for user or MR30)
      .lword dummy_int          ; vector 33 (for user or MR30)
      .lword dummy_int          ; vector 34 (for user or MR30)
      .lword dummy_int          ; vector 35 (for user or MR30)
      .lword dummy_int          ; vector 36 (for user or MR30)
      .lword dummy_int          ; vector 37 (for user or MR30)
      .lword dummy_int          ; vector 38 (for user or MR30)
      .lword dummy_int          ; vector 39 (for user or MR30)
      .lword dummy_int          ; vector 40 (for user or MR30)
      .lword dummy_int          ; vector 41 (for user or MR30)
      .lword dummy_int          ; vector 42 (for user or MR30)
      .lword dummy_int          ; vector 43 (for user or MR30)
      .lword dummy_int          ; vector 44 (for user or MR30)
      .lword dummy_int          ; vector 45 (for user or MR30)
      .lword dummy_int          ; vector 46 (for user or MR30)
      .lword dummy_int          ; vector 47 (for user or MR30)
      ;
;=============================================================
; fixed vector section
;-------------------------------------------------------------
      .section     fvector                  ; fixed vector table
;=============================================================
; special page definition
;-------------------------------------------------------------
; Special page functions can be specified     using
; "#pragma SPECIAL" directive and the macro defined above.
; Uncomment the proper line below to call the macro.
```

```
; See NC30 manual for more information.
;------------------------------------------------------------------
;      SPECIAL 255
;      SPECIAL 254
;      SPECIAL 253
;         :
;         :
;        etc
;         :
;         :
;      SPECIAL 24
;      SPECIAL 23
;      SPECIAL 22
;      SPECIAL 21
;      SPECIAL 20
;      SPECIAL 19
;      SPECIAL 18
;
;==================================================================
; fixed vector section. The 7 or'ed values below (commented out) are for
; specifying the ID codes for serial I/O flash programming
; (highest 8 bits of the vectors). See data sheets for
; more information. Current setting = all zeros by default.
; The highest 8 bits of the reset vector is the parallel protection
; 'register'. Caution! Setting these codes could result in loss of
; all flash programming. See M30624 data sheets before operating
; on these values.
;------------------------------------------------------------------
       .org   0fffdch
UDI:
       .lword dummy_int ;  | 0ff000000h
OVER_FLOW:
       .lword dummy_int ;  | 0ff000000h
BRKI:
       .lword dummy_int
ADDRESS_MATCH:
       .lword dummy_int ;  | 0ff000000h
SINGLE_STEP:
       .lword dummy_int ;  | 0ff000000h
WDT:
       .lword dummy_int ;  | 0ff000000h
DBC:
       .lword dummy_int ;  | 0ff000000h
NMI:
       .lword dummy_int ;  | 0ff000000h
RESET:
       .lword start ;      | 0ff000000h
;
```

RENESAS
Renesas Technology Corp.

Keep safety first in your circuit designs!

- Mitsubishi Electric Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always a possibility of trouble occurring. Trouble with semiconductors may lead to personal injury, fire, or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of backup, auxiliary circuits, (ii) use of non-flammable material, or (iii) prevention against any malfunction or mishap.

Notes regarding this information

- This information is intended as a reference to assist our consumers in the selection of the Mitsubishi semiconductor product best suited to the customer's application; it does not convey any license under any intellectual property rights, or any other rights, belonging to Mitsubishi Electric Corporation or a third party.
- Mitsubishi Electric Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms, represents information on products at the time of publication of these materials, and is subject to change by Mitsubishi Electric Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for the latest product information before purchasing a product listed herein.
- The information described herein may contain technical inaccuracies or typographical errors. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability, or other loss arising from these inaccuracies or errors.
- Please also pay attention to information published by Mitsubishi Electric Corporation by various means, including the Mitsubishi Semiconductor home page (http://www.mitsubishichips.com).
- When using any or all of the information contained in these materials, including product, data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability, or other loss resulting from the information contained herein.
- Mitsubishi Electric Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Mitsubishi Electric Corporation is necessary to reprint or reproduce these materials in whole or in part.
- If these products or technologies are subject to Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
- Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for further details on these materials or the products contained therein.

RENESAS
Renesas Technology Corp.

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

USA Headquarters

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

1050 East Arques Avenue
Sunnyvale, CA 94085-4601
Phone: 408-730-5900
FAX: 408-732-9382

Direct Sales Offices – USA

Northwest

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

1050 East Arques Avenue
Sunnyvale, CA 94085-4601
Phone: 408-730-5900
FAX: 408-737-1129

Southwest

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

20 Fairbanks, Suite 181
Irvine, CA 92618
Phone: 949-859-9453
FAX: 949-859-9450

Rocky Mountain

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

PO Box 273377
Fort Collins, CO 80527
Phone: 970-669-4068
FAX: 970-669-4129

North Central

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

3300 Edinborough Way, Suite 511
Edina, MN 55435
Phone: 952-837-9053
FAX: 952-837-9059

Mitsubishi Electric & Electronics USA, Inc.
(Automotive Market Only)

50 West Big Beaver Rd., Suite 136
Troy, MI 48084
Phone: 248-526-9580
FAX: 248-526-9583

South Central

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

8310 Capital of Texas Hwy. N.,
Suite 260
Austin, TX 78731
Phone: 512-346-4200
FAX: 512-346-4434

Northeast

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

92 Montvale Avenue, Suite 2500
Stoneham, MA 02180
Phone: 781-245-6956
FAX: 781-245-4233

Mid-Atlantic

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

92 Montvale Avenue, Suite 2500
Stoneham, MA 02180
Phone: 781-245-6956
FAX: 781-245-4233

South Atlantic

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

2625 Meridian Parkway, Suite 100
Durham, NC 27713
Phone: 919-767-7900
FAX: 919-767-7902

Southeast

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

2810 Premiere Parkway, Suite 400
Duluth, GA 30097
Phone: 678-258-4518
FAX: 678-258-4519

Mexico Inquiries

Western Mexico

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

20 Fairbanks, Suite 181
Irvine, CA 92618
Phone: 949-859-9453
FAX: 949-859-9450

Eastern Mexico

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

8310 Capital of Texas Hwy. N.,
Suite 260
Austin, TX 78731
Phone: 512-346-4200
FAX: 512-346-4434

Puerto Rico Inquiries

Puerto Rico

Mitsubishi Electric & Electronics USA, Inc.
Electronic Device Group

2810 Premiere Parkway, Suite 400
Duluth, GA 30097
Phone: 678-258-4518
FAX: 678-258-4519

Mitsubishi Electric Sales Canada, Inc.

Direct Sales Offices – Canada

Central & Western Canada

Mitsubishi Electric Sales Canada, Inc.

4299 14th Avenue
Markham, Ontario,
Canada L3R 0J2
Phone: 905-475-7728
FAX: 905-475-1918

Eastern Canada

Mitsubishi Electric Sales Canada, Inc.

340 March Road, Suite 300
Kanata, Ontario,
Canada K2K 2E4
Phone: 613-591-3348
FAX: 613-591-3948

JQA-QS0161

Mitsubishi Electric MCU products sold into the Automotive marketplace are QS-9000 certified.

Printed in U.S.A.

www.mitsubishichips.com