RX コンパイラ CC-RX V2.08.00 リリースノート

R20UT4230JJ0100
Rev.1.00
2017.12.20

この添付資料では，本製品をお使いいただく上での制限事項および注意事項等を記載しております。ご使用の前に，必ずお読みくださいますようお願い申し上げます。

目次

第1章　変更点

本章では，CC-RX のV2.07.00からV2.08.00の変更点について説明します。
以降、professional版のライセンス登録時のみ使用できる機能は【professional】と明記します。

1.1 不正な間接関数呼び出し検出機能の追加【professional】

不正なアドレスへの間接関数呼び出しを検出するための機能を追加しました。
下記の手順で間接関数呼び出しの分岐先アドレスをチェックし、問題を検出した場合にエラー関数を呼び出します。

(1) コンパイル時に、間接関数呼び出しされる可能性のある関数を C ソース・プログラムから自動で抽出し、リンク時にその情報を統合して関数リストを実行形式ファイル内に生成します。

(2) C ソース・プログラムを解析して間接関数呼び出しが行われる直前に、チェック関数”__control_flow_integrity”を呼び出す処理を挿入します。このチェック関数には、当該間接関数呼び出しによる分岐先アドレスが引数として渡されます。

(3) 実行時にチェック関数内で、引数の分岐先アドレスが関数リストに含まれるかをチェックします。含まれていない場合は不正な間接関数呼び出しと判断してエラー関数”__control_flow_chk_fail”を呼び出します。

下記の C ソースを例に説明します。

<ソースコード例>

```
1:extern void func1(void);
2:extern void func2(void);
3:
4:void (*fp)(void) = &func1;
5:
6:void main(void) {
7:    (*fp)();          // 関数func1の間接呼び出し
8:    func2();          // 関数func2の直接呼び出し
9:}
```

4 行目で関数 func1 のアドレスが取得されているため、間接呼び出しされる可能性のある関数と判断して関数リストに func1 を追加します。

7 行目で関数ポインタ fp を使用して間接呼び出ししているため、この呼び出しの直前で関数ポインタ fp の値を取得し、取得した値を引数としてチェック関数”__control_flow_integrity”を呼び出すコードを生成します。チェック関数内では、引数で指定した値（正常に実行している場合は func1 のアドレス）

が関数リストに含まれるかをチェックし、次のように動作します。

- 含まれている場合 ⇒ C ソース・プログラムの処理を続行します。
- 含まれていない場合 ⇒ エラー関数”__control_flow_chk_fail”を呼び出します。

上記のように、不正な間接関数呼び出しを検出することが可能となります。
8 行目は関数 func2 の直接呼出しのため、検出機能の対象外です。

本機能を有効にするには、下記のオプションを指定してください。

【コンパイル・オプション】

```
-control_flow_integrity
```

不正な間接関数呼び出しを検出するコードを生成します。

【リンク・オプション】

```
-cfi
```

不正な間接関数呼び出し検出時に用いる関数リストを生成します。

また、本機能に関連して下記のリンク・オプションも追加しました。

- **-cfi_add_func**
  引数に指定した関数のシンボルまたはアドレスを関数リストに追加します。
- **-cfi_ignore_module**
  引数に指定したファイルに含まれる関数のアドレスを関数リストへ追加しません。
- **-show=cfi**
  -list オプションを指定して出力するリスト・ファイルに関数リストの内容を出力します。

1.2 消費電流チューニングソリューションへの対応

CS+及びe^2 studioの消費電流チューニングソリューションのためのオブジェクトを生成するオプション -insert_nop_with_labelを追加しました。本機能はCS+またはe^2 studio経由での使用が前提であり、ユーザは直接使用しません。
本ソリューションは、E2エミュレータを使用するだけで、ダイナミックに変化するユーザシステムの消費電流を簡単に測定することが可能です。

1.3 メッセージレベル制御機能の追加

警告及びインフォメーションレベルメッセージを抑止するための-no_warningオプションを追加しました。

1.4 ヘキサ・ファイルのレコード長固定機能

常に一定のレコード長でヘキサファイルを出力するための-fix_record_length_and_alignオプションを追加しました。

これによりヘキサファイルの比較等の作業効率が改善します。本機能追加に伴い、-byte_countオプションの仕様を拡張しました。具体的には、-form=stype指定時に-byte_countオプションを指定可能になりました。

1.5 最適化の強化

CC-RX V2.08.00 では、以下(a)、(b)、(c)の最適化強化を実施しました。

(a) -alias=ansiオプション指定時の別名解析の精度改善

別名解析（ソース記述型情報を用いた別名解析）の最適化を改善しました。別名解析（ソース記述型情報を用いた別名解析)は V2.07.00 で実装し、-alias=ansi オプション指定時に有効になる最適化です。V2.07.00 では-merge_files オプション指定時は無効でしたが、V2.08.00 では-merge_files オプション指定時にも有効になるように改善しました。別名解析の最適化が有効になったときに現れる効果は V2.07.00 と同じです。

(b) ビット演算の改善

ビット幅の狭いデータに対するビット演算を改善しました。

```
// ソースコード例
void func(unsigned char *t, unsigned char i, unsigned char j,
          unsigned char v) {
    unsigned char *p = &t[i & 0xff];
    unsigned char m = j >> (i & 0xf);
    *p = v ? m : ~m;
}
```

```
// V2.07.00 の生成コード
_func:
           .STACK      _func=4
           MOV.L R2, R14
           AND #0FH, R14
           SHLR R14, R3
           CMP #00H, R4
           MOVU.B R3, R14
           BNE L12
L11: ; bb20
           OR #0FFFFFF00H, R3
           XOR #0FFH, R3
           MOV.L R3, R14
L12: ; bb23
           MOV.B R14, [R2,R1]
           RTS
```

```
// V2.08.00 の生成コード
_func:
           .STACK      _func=4
           MOV.L R2, R14
           AND #0FH, R14
           SHLR R14, R3
           CMP #00H, R4
           BNE L12
L11: ; bb20
           NOT R3
L12: ; bb23
           MOV.B R3, [R2,R1]
           RTS
```

(c) 冗長な符号拡張の削除

冗長な符号拡張のコードを生成しないようにしました。

```
// ソースコード例
signed char func(signed char * ptr, int n, signed int x) {
  signed int i, j;
  signed char t;

  for (i = 1; i <= n-1; ++i) {
    for ( j = 0; j < n-i; ++j) {
      if ((ptr[j]!=0) && (ptr[j] > ptr[j+1])) {
        t = ptr[j];
        ptr[j] = ptr[j+1];
        ptr[j+1] = t;
      }
    }
  }
  return ptr[x];
}
```

```
// V2.07.00 の生成コード
_func:
                .STACK  _func=16
                PUSHM R6-R8
                SUB #01H, R2
                MOV.L #00000001H, R14
                MOV.L R2, R15
L11:  ; bb59
                CMP R2, R14
                BGT L19
L12:  ; bb59.bb48_crit_edge
                MOV.L #00000000H, R5
                MOV.L R1, R4
L13:  ; bb48
                CMP R15, R5
                BGE L18
L14:  ; bb1
                MOVU.B [R4], R6
                CMP #00H, R6
                BEQ L17
L15:  ; bb8
                MOV.B 01H[R4], R7
                MOV.B R6, R8
                CMP R7, R8
                BLE L17
L16:  ; if_then_bb
                MOV.B R7, [R4]
                MOV.B R6, 01H[R4]
L17:  ; if_break_bb
                ADD #01H, R5
                ADD #01H, R4
                BRA L13
L18:  ; bb56
                SUB #01H, R15
                ADD #01H, R14
                BRA L11
L19:  ; bb66
                MOV.B [R1,R3], R1
                RTSD #0CH, R6-R8
```

```
// V2.08.00 の生成コード
_func:
                .STACK  _func=12
                PUSHM R6-R7
                SUB #01H, R2
                MOV.L #00000001H, R14
                MOV.L R2, R15
L11:  ; bb59
                CMP R2, R14
                BGT L19
L12:  ; bb59.bb48_crit_edge
                MOV.L #00000000H, R5
                MOV.L R1, R4
L13:  ; bb48
                CMP R15, R5
                BGE L18
L14:  ; bb1
                MOV.B [R4], R6
                CMP #00H, R6
                BEQ L17
L15:  ; bb8
                MOV.B 01H[R4], R7
                CMP R7, R6
                BLE L17
L16:  ; if_then_bb
                MOV.B R7, [R4]
                MOV.B R6, 01H[R4]
L17:  ; if_break_bb
                ADD #01H, R5
                ADD #01H, R4
                BRA L13
L18:  ; bb56
                SUB #01H, R15
                ADD #01H, R14
                BRA L11
L19:  ; bb66
                MOV.B [R1,R3], R1
                RTSD #08H, R6-R7
```

1.6 その他の改善点

主に以下の改善を行いました。

(a) 注意事項の改修

switch 文中のラベルへの goto 文を使用している場合の注意事項 (CCRX#045)

(b) CC-RXが使用できるホストPC上のメモリ量を拡張しました。

V2.07.00以前：常に2Gバイトまで

V2.08.00以降：PCが32ビットOSである場合、3Gバイトまで

PCが64ビットOSである場合、4Gバイトまで

(c) 内部エラーの改善

ビルド時に内部エラーが発生する場合がありましたが、これを改善しました。

第2章　注意事項

本章では，CC-RX の注意事項について説明します。

2.1 W0523041メッセージが出力される場合の注意事項 [C/C++コンパイラ]

C 標準ヘッダをインクルードしたファイルを C++または EC++コンパイルしたとき、int_to_short オプションを指定すると W0523041 メッセージが出力されることがあります。この場合は動作には問題ありませんので無視してください。

【注意】
C++および EC++コンパイル時は、int_to_short オプションの指定は無効になります。
C と C++(EC++)との間で共通にアクセスするデータは、int 型ではなく long 型または short 型で宣言してください。

2.2 MVTC,POPC命令を使用する場合の注意事項 [アセンブラ]

アセンブリ言語において、MVTC,POPC 命令に対してプログラムカウンタ(PC)は指定できません。

2.3 deleteオプションをリンク時に指定する場合の注意事項 [最適化リンケージエディタ]

delete オプションで指定した関数シンボルが削除された場合、削除された関数定義の次の関数定義の関数名に対して、デバッグ時にエディタ上でブレークポイントを設定することができません。ラベルウィンドウからブレークポイントを設定するか、関数のプログラム行で指定してください。

第3章　制限事項

本章では，CC-RX の制限事項について説明します。

3.1 C++言語(EC++含む)でmath.hの一部関数(frexp,ldexp,scalbn およびremquo)を使用する場合の制限事項

C++/EC++コンパイル時に、math.h の一部の関数(frexp,ldexp,scalbn,remquo)の実引数を int 型にすると、実行時に無限ループとなるオブジェクトが生成されます。

発生条件:

次の条件(1)(2)を全て満たす場合が該当します。

(1) C++ソース(拡張子が.cpp)または、-lang=cpp オプションが有効である。

(2) math.h をインクルードして、以下の関数をそれぞれの条件で呼び出している。

(a) frexp(double, long *) の第 2 引数の値を (int *)型とする

ただし、第 1 引数が float 型で、-dbl_size=8 オプション指定時を除く

(b) ldexp(double, long) の第 2 引数の値を int 型とする

ただし、第 1 引数が float 型で、-dbl_size=8 オプション指定時を除く

(c) scalbn(double, long) の第 2 引数の値を int 型とする

ただし、第 1 引数が float 型で、-dbl_size=8 オプション指定時を除く

(d) remquo(double, double, long *) の第 3 引数の値を (int *)型とする

ただし、第 1 または第 2 引数が float 型で、-dbl_size=8 オプション指定時を除く

発生例:

[file.cpp]

```
// C++ソースとしてコンパイルした場合に無限ループになる例
#include <math.h>
double d1,d2;
int i;
void func(void)
{
 d2 = frexp(d1, &i);
}
```

[コマンドライン例]

```
ccrx -cpu=rx600 -output=src file.cpp
```

[file.src] ソース出力例

```
_func:
; ...(中略)
BSR __$frexp__tm__2_f__FZ1ZPi_Q2_21_Real_type__tm__4_Z1Z5_Type ; frexp の代替関数を呼ぶ
; ...(中略)
```

```
__$frexp__tm__2_f__FZ1ZPi_Q2_21_Real_type__tm__4_Z1Z5_Type:
L11:
BRA L11 ; 再帰呼び出しとなってしまう
```

回避策:

次のいずれかの方法で回避できます。

(1) -lang=c を指定し、C 言語としてコンパイルする。

(2) 引数の int および int*を long および long*に変更する。

(3) math.h の後に、使用する関数ごとに定義を追加する。

```
/* frexp 関数の場合 */
static inline double frexp(double x, int *y)
{ long v = *y; double d = frexp(x,&v); *y = v; return (d); }
/* ldexp 関数の場合 */
static inline double ldexp(double x, int y)
{ long v = y; double d = ldexp(x,v); return (d); }
/* scalbn 関数の場合 */
static inline double scalbn(double x, int y)
{ long v = y; double d = scalbn(x,v); return (d); }
/* remquo 関数の場合 */
static inline double remquo(double x, double y, int *z)
{ long v = *z; double d = remquo(x,y,&v); *z = v; return (d); }
```

回避策(2)の例

[file.cpp] の変更例

```
#include <math.h>
double d1,d2;
int i;
void func(void)
{
   long x = i; /* 一旦 long 型変数で受ける */
   d2 = frexp(d1, &x); /* long 型変数で呼び出し */
   i = x; /* i に値を設定 */
}
```

回避策(3)の例

[file.cpp] の変更例

```
#include <math.h>
/* 宣言を追加 */
static inline double frexp(double x, int *y)
{ long v = *y; double d = frexp(x,&v); *y = v; return (d); }
double d1,d2;
int i;
void func(void)
{
   d2 = frexp(d1, &i);
}
```

3.2 PIC/PID機能ご利用に関する制限事項 (picおよびpidオプション)

ライブラリジェネレータ lbgrx に pic または pid オプションを指定して、標準ライブラリを作成することができますが、その際に、次の警告が 1 回または複数回表示されます。

```
W0591301:"-pic" option ignored (pic オプションを指定した場合)
W0591301:"-pid" option ignored (pid オプションを指定した場合)
```

これらの警告は、EC++ライブラリに対して、pic, pid オプションが無効になるため出力されます。

3.3 以下のオプションを廃止しました【C/C++コンパイラ】

(a) -file_inline, -file_inline_path

指定しても警告を表示し無効となります。代替手段としては、ソース中に#includeを記述してください。C(C99含む)の場合は、同様の機能を持つmerge_filesも使用できます。

(b) -enable_register

指定しても無視されます。指定の有無による生成コードの変化はありません。

3.4 C/C++ソースレベルデバッグに関する注意事項【C/C++コンパイラ】

(1)-debugオプションを指定しても､次の行に対しては､デバッガでブレークポイントの設定やステップ実行で停止ができない場合があります。

・グローバル変数の動的な初期化式(C++)

・次のような関数の先頭行

#pragma inline_asm が指定されている関数。

ループ文(do文,while文など)から始まり、かつauto変数を持たない関数。

・ループ(for文, while文,do文)の制御式と本体を1行で記述した場合の制御式および本体部分。

(2)共用体型かつレジスタ渡しである仮引数のメンバが、ウォッチウィンドウ等で誤った値が表示される場合があります。

3.5 0xffffffff番地を含むセクションを記述する場合の注意事項【アセンブラ】

アセンブリソース中に同じセクション名に対する.org 制御命令を含む.section 制御命令が複数あり、これらが互いに 0xffffffff 番地で重複している場合、アセンブル時に内部エラー(C0554098)が発生します。

例)

```
.section SS,ROMDATA
.org 0fffffffeh
.byte 1
.byte 2 ; 0xffffffff
.section SS,ROMDATA
.org 0ffffffffh
.byte 3; ; 0xffffffff
.end
```

3.6 -formと-outputオプションを同時に使用する場合の注意事項【リンカ】

リンカ(rlink)に、-form=rel と-output=出力ファイル名を共に指定したとき、出力ファイル名の拡張子が無視され、常に.rel に置き換わります。

例) rlink -form=relocate -output=DefaultBuild¥lib_test.lib

出力ファイルを test.lib とすべきところが test.rel になります。

3.7 _builtinから始まる関数名を使用する場合の注意事項【C/C++コンパイラ】

include ディレクトリ内の machine.h に記述のある _builtin から始まる関数名をソースコードで宣言した場合、内部エラーになることがあります。アンダーバー(_)から始まる名前は予約されていますので、お客様のソースコードには記述しないでください。

3.8 save_accが有効な#pragma interrupt指定された関数が仮引数を持つ場合の注意事項【C/C++コンパイラ】

#pragma interrupt で指定された関数で、save_acc フラグが有効(-save_acc オプション指定時を含む)な場合、生成コードが R4 で渡される仮引数を正常に取得できない場合があります。

注意) #pragma interrupt で指定された関数に仮引数を定義することは、推奨しておりません。

3.9 -merge_filesの制限事項

共用体型変数を記述した翻訳単位を -merge_files を指定してコンパイルすると F0530800 が発生する制限事項

[発生条件]

次の条件を全て満たす場合，F0530800 または W0530811 が発生する可能性があります。

(1) -merge_files または -whole_program を指定している。

(2) 2 つ以上のメンバを持つ共用体型外部変数を初期化しており，かつ初期化対象のメンバ以外に，アライメントとサイズの両方が他のいずれのメンバよりも大きいメンバが存在する。

(3) (2) に該当する変数を次のいずれかで extern 宣言し，参照している。

(3-1) (2) の外部変数定義が存在するソースファイルと別のソースファイル。

(3-2) (2) の外部変数定義が存在するソースファイルと別のソースファイルから，直接，または間接的にインクルードされているヘッダファイル。

[回避策]

次のいずれかの対応を実施してください。

(1) 発生条件 (1) のオプションをいずれも指定しない。

(2) 発生条件 (2) における "初期化" を関数内で実施する。

(3) 発生条件 (2) に該当する変数を，その外部変数定義を記述したソースファイル内でのみ参照する。

3.10 -cfi_ignore_module オプションの制限事項

C/C++ソースファイルを-output=absを使用してコンパイルした時、生成されるオブジェクトファイルを-cfi_ignore_moduleで指定できません。

-output=objを使用してオブジェクトファイルを生成し、その生成したオブジェクトファイルを-cfi_ignore_moduleで指定してください。

第4章　添付標準ライブラリについて

本章では，RXファミリC/C++コンパイラ 添付標準ライブラリについて説明します。

本製品では、RX600 用に標準ライブラリファイル(*.lib)を 4 種類添付しています。
添付の標準ライブラリファイルを使用することにより、ビルドに要する時間を短縮することができます。

4.1 添付ライブラリー覧

本製品に添付される、標準ライブラリファイルの一覧を表 4.1 に示します。

【ご注意】
ライブラリで選択している「マイコンオプション」は、ご使用のコンパイラオプションと一致させる必要があります。いずれとも一致しない場合は、これらの標準ライブラリは使用できませんので、ご利用のコンパイラオプション(アセンブラオプション)をライブラリジェネレータに指定して、作成されるライブラリをご使用ください。

<table>
<tr><th rowspan="3">ライブラリ名</th><th rowspan="3">用途</th><th rowspan="3">最適化*2
オプション</th><th colspan="3">マイコンオプション *1 *2</th></tr>
<tr><th rowspan="2">-endian</th><th>-cpu</th><th rowspan="2">その他 *3</th></tr>
<tr><th>-rtti
-exception
-noexception</th></tr>
<tr><td>rx600lq.lib</td><td>RX600、速度優先
リトルエンディアン</td><td>-speed
-goptimize</td><td rowspan="2">-endian=little</td><td rowspan="4">-cpu=rx600
-rtti=on
-exception</td><td rowspan="4">-round=nearest
-denormalize=off
-dbl_size=4
-unsigned_char
-unsigned_bitfield
-bit_order=right
-unpack
-fint_register=0
-branch=24</td></tr>
<tr><td>rx600ls.lib</td><td>RX600、サイズ優先
リトルエンディアン</td><td>-size
-goptimize</td></tr>
<tr><td>rx600bq.lib</td><td>RX600、速度優先
ビッグエンディアン</td><td>-speed
-goptimize</td><td rowspan="2">-endian=big</td></tr>
<tr><td>rx600bs.lib</td><td>RX600、サイズ優先
ビッグエンディアン</td><td>-size
-goptimize</td></tr>
</table>

表 4.1 ライブラリー覧

*1　マイコンオプションは、ユーザーズマニュアル RX ビルド編の「A.1.3 オプション」「(1)コンパイル・オプション」「 マイコンオプション」を参照ください。

*2　これらのオプション選択は省略時設定と同じです。

4.2 ライブラリ指定の方法

コンパイラパッケージに含むライブラリファイルは、セクション 4.2.1 および 4.2.2 で示した方法でリンクしてください。

4.2.1 ライブラリ指定の方法

e² studio を C:¥Renesas¥e2_studio へインストールした時、ライブラリファイルは、以下のフォルダに存在します。

```
C:¥Program Files¥Renesas¥RX¥V2_8_0¥lib
```

（「V2.08.00」は、コンパイラパッケージのバージョンおよび改訂番号を示します。）

4.2.2 最適化リンケージエディタの中でライブラリファイルのフォルダを指定

コンパイラパッケージに含まれたライブラリファイル(セクション4.2.1で示した位置に格納されたファイル)を希望のディレクトリにコピーしてください。
次に、コピーしたライブラリファイルのうちの1つを指定してリンケージ処理を始めてください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、その他の不適切に使用しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   標準水準：　コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等

   高品質水準：　輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、
   金融端末基幹システム、各種安全制御装置等

   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することはできません。たとえ、意図しない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を、(1)核兵器、化学兵器、生物兵器等の大量破壊兵器およびこれらを運搬することができるミサイル（無人航空機を含みます。）の開発、設計、製造、使用もしくは貯蔵等の目的、(2)通常兵器の開発、設計、製造または使用の目的、または(3)その他の国際的な平和および安全の維持の妨げとなる目的で、自ら使用せず、かつ、第三者に使用、販売、譲渡、輸出、賃貸もしくは使用許諾しないでください。

   当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様の転売、貸与等により、本書（本ご注意書きを含みます。）記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は一切その責任を負わず、お客様にかかる使用に基づく当社への請求につき当社を免責いただきます。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載された情報または当社製品に関し、ご不明点がある場合には、当社営業にお問い合わせください。

注1.　本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注2.　本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

(Rev.3.0-1　2016.11)

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口　http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社　〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：https://www.renesas.com/contact/