CubeSuite+ RX ファミリ コンパイラ CC-RX V1.02.01 リリースノート

R20UT1993JJ0100
Rev.1.00
2012.3.14

この度は，統合開発環境 CubeSuite+をご使用いただきまして，誠にありがとうございます。

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　対象デバイスについて

CC-RX がサポートする対象デバイスに関しては，WEB サイトに掲載しています。

こちらをご覧ください。

CubeSuite+製品ページ：

http://japan.renesas.com/cubesuite+

第2章　アンインストール時の選択キーワード

本製品をアンインストールする場合は，2 つの方法があります。

・統合アンインストーラを使用する(CubeSuite+自体をアンインストールする)

・個別にアンインストールする(本製品のみをアンインストールする)

個別にアンインストールを行なう場合，コントロールパネルの

・「プログラムの追加と削除」(WindowsXP の場合)

・「プログラムと機能」(Windows Vista, Windows 7 の場合)

から，「CubeSuite+ CC-RX V1.02.00」を選択してください。

第3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。

3.1　C1804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 C1804 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。

3.2　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。

アセンブラでの割り込み関数を記述する場合には，r1 レジスタの退避／復帰を行ってください。

3.3　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4章　制限事項

本章では，CC-RX V1.02.00 の制限事項について説明します。

4.1　PID機能ご利用に関する制限事項 (nouse_pid_registerオプション)

PID 機能を利用するとき、マニュアルに従って、マスターに含まれる全てのファイルに nouse_pid_register を指定すると、下記をアセンブルする際にエラーが発生します。

(1) PID レジスタにアドレスを代入するプログラム

(2) 標準ライブラリ(ライブラリジェネレータ lbgrx に指定)

【(1)の制限について】

PID レジスタにアドレスを代入するその機能だけをひとつの C ソースもしくはアセンブラファイルにして、-nouse_pid_register を指定せずにコンパイルまたはアセンブルしてください。そして、他のマスターのファイル(-nouse_pid_register 付き)とリンクして利用してください。

【(2)の制限について】

次の(a)(b)いずれかの方法で対応ください。

(a) 標準ライブラリをマスターではなくアプリケーションに収録する

・ジャンプテーブル(8.4.2(2)参照)を利用する必要はありません。

・ライブラリジェネレータ lbgrx に pid オプションを指定して標準ライブラリを作成してください。

(b)標準ライブラリをマスターに配置する場合

(i) ライブラリジェネレータ lbgrx に nouse_pid_register オプションをつけずに標準ライブラリを作成してください。

(ii) 作成したジャンプテーブルを、次の例に従って、全てのエントリの JMP R14 を変更してから使用してください。

例) _printf エントリの変更

[変更前]

```
_printf:
        MOV.L  #0ffff90cfH,R14 ; アドレス 0ffff90cfH は例です。
        JMP R14
```

[変更後]

```
_printf:
        MOV.L  #0ffff90cfH,R14
        PUSH.L R13 ; PID レジスタが R13 の場合
        JSR    R14
        POP    R13 ; PID レジスタが R13 の場合
        RTS
```

4.2 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);
}
```

4.3 標準ライブラリのご利用に関するご注意 【V1.02.00 からリビジョンアップされたお客様】

V1.02.00 で標準ライブラリをご利用のお客様の場合、V1.02.01 以降にリビジョンアップ後は、環境変数 TMP_RX を V1.02.00 とは異なるディレクトリに変更してご利用いただくようお願いします。これは、ライブラリジェネレータ lbgrx は、ライブラリ作成時の中間結果を TMP_RX が示すディレクトリに保存し、次回のライブラリ作成時に再利用するためです。

この対応を行わない場合、lbgrx が生成する標準ライブラリが、リビジョンアップした環境で生成したライブラリになりません。

なお、統合環境 CubeSuite+をご利用の場合は、次の対応を行ってください。

【統合環境(CubeSuite+)で標準ライブラリをご利用の場合の対応手順】

次の(1)～(3)の手順を、V1.02.01 にリビジョンアップ後に 1 回実施ください。

(1) コマンドプロンプトを開きます。(以降、コマンドプロンプト上で作業を行います。)

(2) コマンドラインで dir %TEMP%¥*.pgl を実行し、次のような、数字とアルファベットの並びを持ち、かつ拡張子が.pgl というファイルがひとつまたは複数表示されることを確認してください。

例) dir %TEMP%¥.pgl の表示結果

```
2011/08/09 15:47 825,346 8000040080100000225a40409694ab0200000000.pgl
```

(3)で表示されたファイルの個数だけ、次のように del コマンドを使ってファイルをひとつずつ削除してください。

例) ファイルを削除するコマンドの例(1 ファイル分)

```
del %TEMP%¥8000040080100000225a40409694ab0200000000.pgl
```

[ご参考]

手順(2)で、拡張子.pgl のファイルが手順(2)の例で示した形式以外に表示されていなければ、次のように一括して削除することもできます。

例) ファイルを一括で削除するコマンドの例

```
del %TEMP%¥*.pgl
```

4.4 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。

```
C1301 (W) "-pic" option ignored (pic オプションを指定した場合)
C1301 (W) "-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。

4.5 旧バージョン（V1.00.xx）から移行したプロジェクトのビルドについて --（Lセクションの警告L1120 およびエラーL3100）

V.1.01 よりも古いコンパイラパッケージで作成した High-performance Embedded Workshop のプロジェクトを、V1.02.00 のプロジェクトに変換して使用すると、次の警告およびエラーを発生することがあります。

```
L1120 (W) Section address is not assigned to "L"
L3100 (F) Section address overflow out of range : "L"
```

この場合は、本書「5.ヘルプ、マニュアル修正項目」の「■ 9.4.1 V.1.00 との互換性 / (2) L セクション追加について(section オプション、Start オプション)」の内容に基づき、(a)または(b)の方法で対策してください。

【備考】

本パッケージに含まれる CubeSuite+で、V.1.01 よりも古いプロジェクト（RX Toolchain 1.0.0.0 ～ 1.0.0.2）を V1.02.00 またはそれ以降に変換した場合は、自動的に(b)を行いますので、この現象は発生しません

第5章　ヘルプ、マニュアル修正項目

本章では，CC-RX V1.02.01 のヘルプおよびマニュアルの修正項目を示します。

■ コーディング編 / 3.2.3 #pragma 指令 / 表 3-12 #pragma 指令リスト

＜追加内容＞

#pragma 指令 /

#pragma STDC CX_LIMITED_RANGE フラグ

#pragma STDC FENV_ACCESS フラグ

#pragma STDC FP_CONTRACT フラグ

用途 / システムの状態変更 (C99 言語を有効にしてコンパイルする場合、これらに対しては C99 言語の文法確認のみ行い、内容は無視します。)

■ コーディング編 / 3.2.4 拡張仕様の使用方法 / (9)構造体/クラスメンバのアライメント指定 (#pragma pack)

＜削除内容＞

#pragma pack を指定した構造体メンバおよびクラスのメンバはポインタを用いてアクセスすることはできません(ポインタを使用したメンバ関数内でのアクセスを含みます)。

例

```
#pragma pack
struct st {
  char x;
  int y;
} ST;
int *p=&ST.y; /* ST.y のアドレスが奇数になる場合があります */
void func(void) {
  ST.y=1; /* 正しくアクセスできます */
  *p=1; /* 正しくアクセスできない場合があります */
}
```

■ コーディング編 / 6.4.11 <stdlib.h> / mbstowcs 関数

＜追加内容＞

定義名 / mbstowcs

多バイト文字列をワイド文字列に変換 (詳細は 6.4.20 wchar.h / mbstowcs を参照)

■ コーディング編 / 6.4.11 <stdlib.h> / wcstombs 関数

＜追加内容＞

定義名 / wcstombs

内容 / 多バイト文字列をワイド文字列に変換 (詳細は 6.4.20 wchar.h / wcstombs を参照)

■ コーディング編 / 9.4.1 V.1.00 との互換性 / (2)L セクション追加について(section オプション、Start オプション)

＜修正内容＞ ※変更後の内容のみ示します。

[変更後]

(2) L セクション追加について(section オプション、Start オプション)

V.1.01 では、C セクションに対する map や base オプション適用時のコード効率向上を図るため、文字列リテラルなどのリテラル領域の出力先として L セクションを新規に追加しました。

L セクションの割り付けアドレスを指定しない V.1.00 からプロジェクト変換したプロジェクトは、リンク時に L セクションに対して、最適化リンケージエディタがアドレスエラーL3100(F)を出力する場合があります。

```
L3100 (F) Section address overflow out of range : "L"
```

これを回避するためには、次のいずれかの方法を実施してください。なお、コード効率の面から、通常は(a)の方法を推奨します。(b)の方法は、セクション構成を変えたくない場合に使用ください。

(a)リンク時の最適化リンケージエディタの Start オプションに指定するセクション列に L を追加する

[コマンドラインでの指定方法]

例)

V.1.00 での指定例

```
-start=B_1,R_1,B_2,R_2,B,R,SU,SI/01000,PResetPRG/0FFFF8000,C_1,
C_2,C,C$*,D*,P,PIntPRG,W*/0FFFF8100,FIXEDVECT/0FFFFFFD0
```

変更例(C の後に L を追加する)

```
-start=B_1,R_1,B_2,R_2,B,R,SU,SI/01000,PResetPRG/0FFFF8000,C_1,
C_2,C,L,C$*,D*,P,PIntPRG,W*/0FFFF8100,FIXEDVECT/0FFFFFFD0
```

CubeSuite+での設定手順は以下のとおりです。

1. CC-RX (ビルド・ツール) を選択する。
2. 「リンク・オプション」タブをクリックし、「セクション」から「セクション開始アドレス」を選択する
3. セクション並びの右端にある[...]ボタンを押す
4. L セクションを配置する Address 領域をリスト表示から選択して「編集」または「追加」ボタンをクリックして、L セクションを配置するアドレスを設定する

(b)コンパイル時およびライブラリビルド時に-section=L=C を選択する

コンパイル時、およびライブラリビルド時に、ccrx および lbgrx コマンドのそれぞれに -section=L=C を指定することで、リテラル領域の出力先が C セクションに変更され、V.1.00 互換のセクション構成にすることができます。

■ ビルド / B.1.3 オプション / (1)コンパイル・オプション / オブジェクトオプション / -nouse_div_inst / [備考]

＜追加内容＞

本オプションは、ライブラリジェネレータ(lbgrx)でも指定することができます。

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、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に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障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。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。お客様が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。
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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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。

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・電話番号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 エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■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：http://japan.renesas.com/inquiry