

【注意事項】

R20TS0373JJ0100

Rev.1.00

RL78 ファミリ用 C コンパイラパッケージ

2018.12.01 号

概要

RL78 ファミリ用 C コンパイラパッケージ CC-RL の使用上の注意事項を連絡します。

1. switch 文の制御式に 1 ビットの符号付きビット・フィールドを記述する場合の注意事項 (CCRL#020)
2. far ポインタのメンバを持つ構造体または共用体をパッキングして far 領域に配置する場合の注意事項(CCRL#021)

注: 注意事項の後ろの番号は、注意事項の識別番号です。

1. switch 文の制御式に 1 ビットの符号付きビット・フィールドを記述する場合の注意事項(CCRL#020)

1.1 該当製品

CC-RL V1.00.00 ~ V1.07.00

1.2 内容

switch 文の制御式に 1 ビットの符号付きビット・フィールドを記述する場合、値が 1^(注1)の時の case ラベルを実行する場合があります。

注 1: 1 ビットの符号付きビット・フィールドの値は 0 または -1 のいずれかとなります。

1.3 発生条件

以下の(1)~(2)の条件をすべて満たす場合に不正なコードを生成する可能性があります。

- (1) switch 文の制御式に 1 ビットの符号付きビット・フィールドを記述する。
- (2) (1)の case ラベルに値が 1 の時の処理を記述する。

1.4 発生例

以下に、発生例を記します。赤文字が発生条件の該当箇所です。

【C ソース】

```

1  typedef struct{
2      signed char b0:1;
3      signed char b1:1;
4  } ST;
5
6  void func1(ST n) {
7      switch (n.b0) {      // 発生条件(1)
8          case 0:
9              func0(0);
10             break;
11             case 1:      // 発生条件(2)
12                 func0(1);
13                 break;
14             default:
15                 break;
16         }
17     }

```

- ・7行目：1ビットの符号付きビット・フィールド n.b0 を switch 文の制御式に記述しているため、条件(1)に該当します。
- ・11行目：case ラベルに値が1の時の処理を記述しているため、条件(2)に該当します。n.b0 の取り得る値は0か-1ですが、出力コードでは n.b0 が-1の時に default: に分岐すべきところを case 1: に分岐します。

【出力アセンブラコード】

```

1  _func1:
2      bt a.0, $.BB@LABEL@1_2 ; ビットが立っていれば case 1: に分岐(不正)
3      .BB@LABEL@1_1:          ; case 0: の処理
4          clrw ax
5          br !!_func0
6      .BB@LABEL@1_2:          ; case 1: の処理
7          onew ax
8          br !!_func0

```

n.b0 の取り得る値は0か-1ですが、n.b0 が-1の時に case 1: に分岐するコードを出力します。

1.5 回避策

下記(1)~(2)のいずれかにより回避可能です。

- (1) 発生条件(1)の switch 文を、if 文に置き換える。
- (2) 発生条件(1)の1ビットのビット・フィールドの型を符号なしに変更する。

1.6 恒久対策

CC-RL V1.08.00 で改修する予定です。

2. far ポインタのメンバを持つ構造体または共用体をパッキングして far 領域に配置する場合の注意事項(CCRL#021)

2.1 該当製品

CC-RL V1.01.00 ~ V1.07.00

2.2 内容

far ポインタのメンバを持つ構造体または共用体をパッキングして far 領域に配置する場合、不正動作となる可能性があります。

2.3 発生条件

以下の(1)~(4)の条件をすべて満たす場合に不正なコードを生成する可能性があります。

- (1) far ポインタのメンバを持つ構造体型または共用体型の変数を定義する。
- (2) (1)の変数を far 領域に配置する。
- (3) (1)の変数に対し構造体のパッキング(-pack オプションまたは#pragma pack)を指定する。
- (4) (1)の変数の far ポインタのメンバをアクセスする。

2.4 発生例

以下に、発生例を記します。赤字が発生条件の該当箇所です。

【C ソース】

| | | |
|---|---------------------------------------|----------------------------|
| 1 | <code>#pragma pack</code> | <code>// 発生条件(3)</code> |
| 2 | <code>struct ST {</code> | |
| 3 | <code>char c;</code> | |
| 4 | <code>int __far *ifp;</code> | <code>// 発生条件(1)</code> |
| 5 | <code>} __far st1;</code> | <code>// 発生条件(1)(2)</code> |
| 6 | | |
| 7 | <code>int __far * func(void) {</code> | |
| 8 | <code>return st1.ifp;</code> | <code>// 発生条件(4)</code> |
| 9 | <code>}</code> | |

- 1 行目：#pragma pack を指定しているため、条件(3)に該当します。
- 4~5 行目：far ポインタのメンバ ifp を持つ構造体型変数 st1 を定義しているため、条件(1)に該当します。また、st1 を__far で修飾しているため、条件(2)に該当します。
- 8 行目：st1 のメンバ ifp を参照しているため、条件(4)に該当します。

【出力アセンブラコード】

| | |
|---|--|
| 1 | _func: |
| 2 | mov es, #LOW(HIGHW(_st1)) |
| 3 | mov a, es:!LOWW(_st1+0x00003) |
| 4 | movw de, es:!LOWW(_st1+0x00001) ; 奇数番地に対する movw 命令 |

st1 のメンバ ifp を参照する際に 16bit 転送命令を出力するため、ifp が奇数番地に配置された場合、奇数番地に対して 16bit 転送命令を実行し不正な動作となります。

2.5 回避策

下記(1)～(2)のいずれかにより回避可能です。

- (1) 発生条件(1)の変数を near 領域に配置する。
- (2) 発生条件(1)の変数に対して構造体のパッキングを指定しない。

2.6 恒久対策

CC-RL V1.08.00 で改修する予定です。

以上

改訂記録

| Rev. | 発行日 | 改訂内容 | |
|------|------------|------|------|
| | | ページ | ポイント |
| 1.00 | 2018.12.01 | - | 新規発行 |
| | | | |

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)

■総合お問い合わせ先

<https://www.renesas.com/contact/>

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。

過去のニュース内容は発行当時の情報をもとにしており、現時点では変更された情報や無効な情報が含まれている場合があります。

ニュース本文中の URL を予告なしに変更または中止することがありますので、あらかじめご承知ください。

すべての商標および登録商標は、それぞれの所有者に帰属します。