

リンク・ディレクティブの説明

RL78用 Cコンパイラ CA78K0R

株式会社ルネサス ソリューションズ
ツールビジネス本部 ツール技術部

2014/6/20 Rev. 1.00

R20UT3045JJ0100

- **リンク・ディレクトィブとは**
- **リンク・ディレクトィブのイメージ**
- **デフォルトのリンク・ディレクトィブのイメージ**
- **セグメントの属性**
- **リンク・ディレクトィブの記述形式**
- **メモリ領域の変更**
- **セグメント配置の変更**
- **Cコンパイラが出力するセグメント名の変更**
- **Cコンパイラで出力されるセグメント名と属性**
- **高度なリンク・ディレクトィブの機能の説明**
- **メモリ空間のイメージ**
- **リンク・ディレクトィブの記述形式(2)**
- **CubeSuite+ 上でのリンク・ディレクトィブ・ファイルの指定方法**
- **CubeSuite+ 上でのリンク・ディレクトィブの生成**

リンク・ディレクティブとは

■ リンク・ディレクティブとは

- リンカに対して入力ファイルや使用可能なメモリ領域、セグメントの配置など、リンク時の各種指示を行うための命令群です。
- セグメントをマイコン上のメモリに、どのように配置するかを指定します
- リンク・ディレクティブ・ファイル(テキスト形式)を作成し、この中に上記の各種指示を記載して、このファイルをリンクに指定します

■ ディレクティブの種類

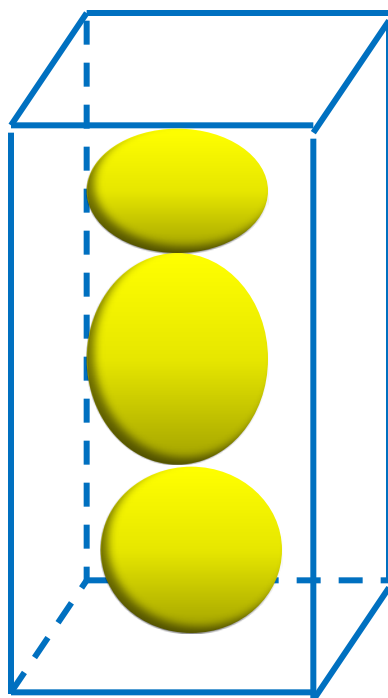
- メモリ・ディレクティブ
 - セグメントを配置するメモリ領域を定義します。
- セグメント配置ディレクティブ
 - 各セグメントの配置を指定します。

■ セクション、セグメントの表現の注意

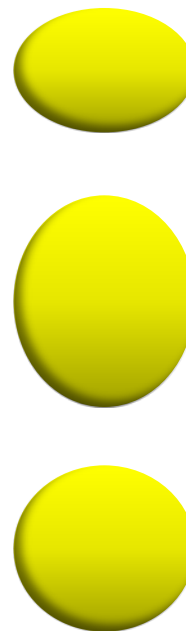
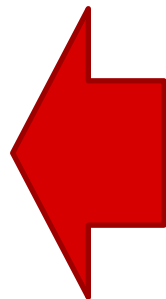
- CA78K0Rでは、Cコンパイラではセクション、アセンブラではセグメントと呼びます。ただし、セクション＝セグメントで、同じものを示しています。

リンク・ディレクティブのイメージ

- マイコンのメモリマップ上に箱(メモリ領域)を用意してから、その中にボール(セグメント)を配置します。
- 箱(メモリ領域)のサイズや位置(アドレス)や、ボール(セグメント)を格納する箱(メモリ領域)や位置(アドレス)を指定します。



箱(メモリ領域)



ボール(セグメント)

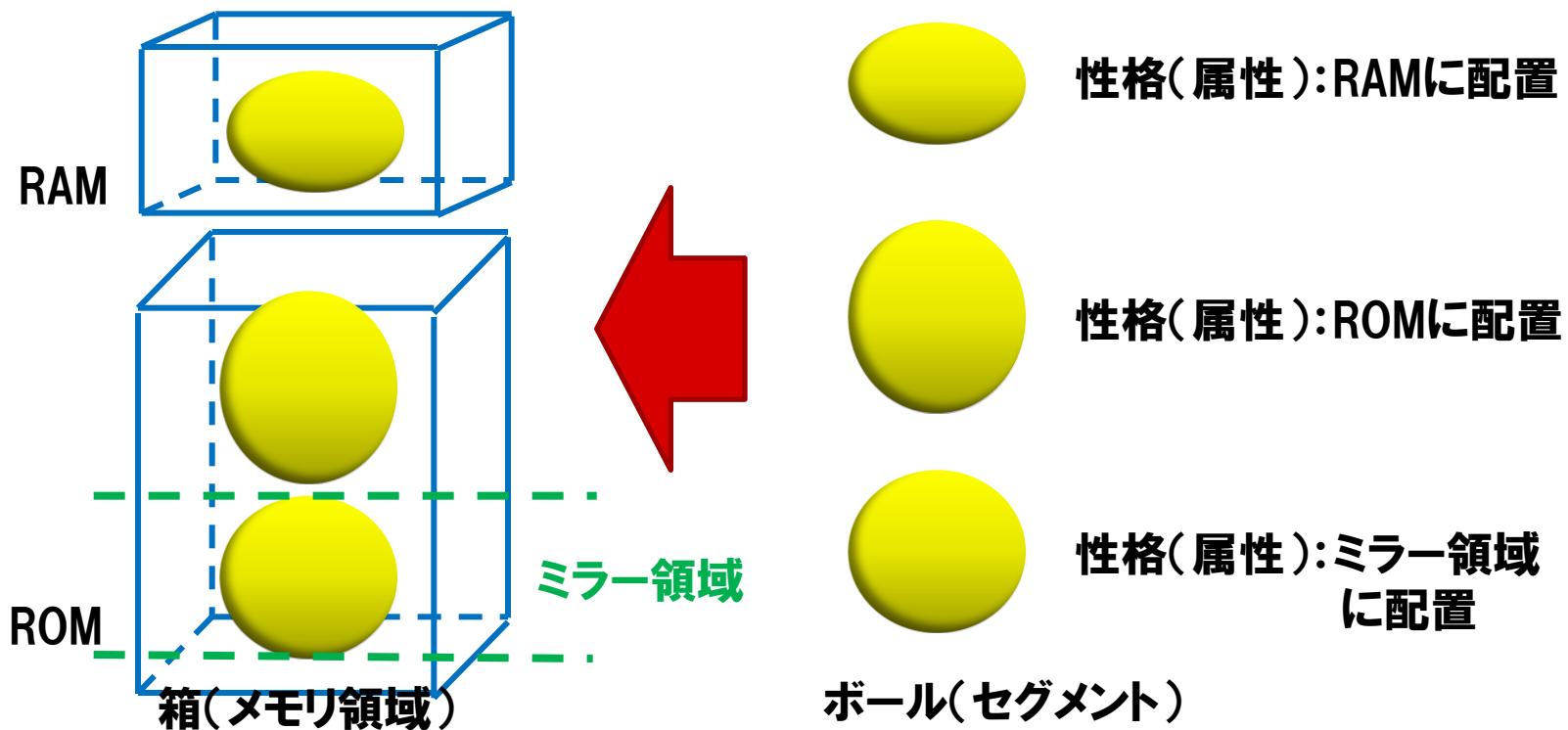
デフォルトのリンク・ディレクティブのイメージ

■ 箱(メモリ領域)

- ROMとRAMが用意されています

■ ボール(セグメント)

- Cコンパイラにより、いろいろな**性格(属性)**のセグメントが生成されます
- **性格(属性)**により、入れられる箱や位置が決まってきます



セグメントの属性(1/3)

■ コードセグメント(ROMに配置されるセグメント)の属性(再配置属性)

再配置属性	説明
CALLTO	指定セグメントを00080H~000BFH番地内で先頭が偶数番地になるように配置します。
FIXED	指定セグメントを000C0H~OFFFHH番地に配置します。
BASE	指定セグメントを000C0H~OFFFHHに配置します。
AT	指定セグメントを絶対番地に配置します(SFR, 2ndSFR領域を除く)。
UNIT	指定セグメントを任意の位置へ配置します(メモリ領域名“ROM”内 000C0H~EFFFFH)。
UNITP	指定セグメントを任意の位置へ、先頭が偶数番地になるように配置します(メモリ領域名“ROM”内 000C0H~EFFFFH)。
IXRAM	指定セグメントを任意の位置へ配置します(メモリ領域名“ROM”内 000C0H~EFFFFH)。
SECUR_ID	セキュリティID指定専用の属性です。セキュリティID以外は指定しないでください。指定セグメントを000C4H~000CDH番地へ配置します。
PAGE64KP	64K境界にまたがらない、先頭が偶数番地となるように配置します。異なるファイルの同名セグメントは結合しません。
UNIT64KP	64K境界にまたがらない、先頭が偶数番地となるように配置します。同名セグメントは結合します。
MIRRORP	MAA=0のときのミラー元領域(01000H~0xxxxH), またはMAA=1のときのミラー元領域(11000H~1xxxxH)のいずれかの領域に配置します。
OPT_BYTE	ユーザ・オプション・バイト, およびオンチップ・デバッグ指定専用の属性です。ユーザ・オプション・バイト, およびオンチップ・デバッグ以外は指定しないでください。指定セグメントを000C0H~000C3H番地へ配置します。

セグメントの属性(2/3)

■ データセグメント(RAMに配置されるセグメント)の属性(再配置属性)

再配置属性	説明
SADDR	指定セグメントをsaddr領域に配置します(saddr領域:FFE20H~FFEFFH)。
SADDRP	指定セグメントをsaddr領域に先頭が偶数番地となるように配置します(saddr領域:FFE20H~FFEFFH)。
AT	指定セグメントを絶対番地に配置します(SFR, 2ndSFR領域を除く)。
UNIT	指定セグメントを内部または外部の任意の位置へ、偶数番地から配置します(メモリ領域名“RAM”内)。
UNITP	指定セグメントを内部RAM領域に先頭が偶数番地となるように配置します(ただし、saddr領域(FxxxxH~FFEFFH)は含みません)。ES参照なしでアクセスしたいデータを配置するときに使用します。
BASEP	指定セグメントを内部RAM領域に先頭が偶数番地となるように配置します(ただし、saddr領域(FxxxxH~FFEFFH)は含みません)。ES参照なしでアクセスしたいデータを配置するときに使用します。
PAGE64KP	メモリ領域名“RAM”内に、64K境界にまたがらない、先頭が偶数番地となるように配置します。異なるファイルの同名セグメントは結合しません。
UNIT64KP	メモリ領域名“RAM”内に、64K境界にまたがらない、先頭が偶数番地となるように配置します。同名セグメントは結合します。

セグメントの属性(3/3)

■ ビットセグメント(RAMに配置されるセグメント)の属性(再配置属性)

再配置属性	説明
AT	指定セグメントを絶対番地に配置します(SFR, 2ndSFR領域を除く)。
UNIT	指定セグメントを内部または外部の任意の位置へ, 偶数番地から配置します(メモリ領域名“RAM”内)。

リンク・ディレクティブの記述形式

■ メモリ・ディレクティブ

- **MEMORY** **メモリ領域名** : (**スタート・アドレス** , **サイズ**)
 - **メモリ領域名**
 - 作成する**箱(メモリ領域)**の名前を指定します
 - **スタート・アドレス**
 - 作成する**箱(メモリ領域)**の開始アドレスを指定します
 - **サイズ**
 - 作成する**箱(メモリ領域)**のサイズを指定します
 - 終了アドレスではないので注意して下さい

■ セグメント配置ディレクティブ

- **MERGE** **セグメント名** : [**AT** (**スタート・アドレス**)] [= **メモリ領域名**]
 - **セグメント名**
 - 配置を指定する**ボール(セグメント)**の名前を指定します
 - **スタート・アドレス**
 - 配置を指定する**ボール(セグメント)**の開始アドレスを指定します(省略可)
 - **メモリ領域名**
 - 配置を指定する**ボール(セグメント)**の格納する**箱(メモリ領域)**を指定します(省略可)

メモリ領域の変更(1/3)

- **箱(メモリ領域)**を変更する場合には、デフォルトのメモリ領域を考慮する必要があります。
 - 内蔵ROM／内蔵RAMに新たな**箱(メモリ領域)**を追加する
 - 新たに確保したい**箱(メモリ領域)**を除いた分で、デフォルトで用意されている**箱(メモリ領域)**のROM、RAMの位置(アドレス)、大きさ(サイズ)を**指定しなおす必要があります**。
 - 外部メモリに新たな**箱(メモリ領域)**を追加する
 - 外部メモリにはデフォルトで用意されている**箱(メモリ領域)**は存在しないので、そのまま外部メモリ用の**箱(メモリ領域)**を定義できます。

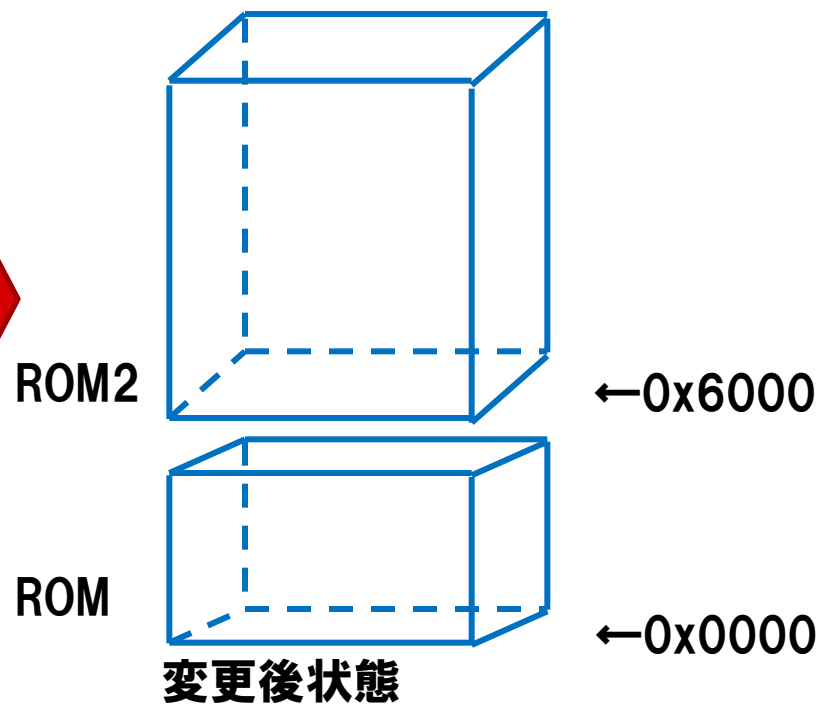
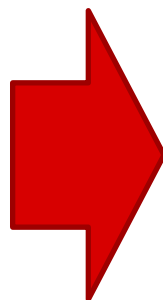
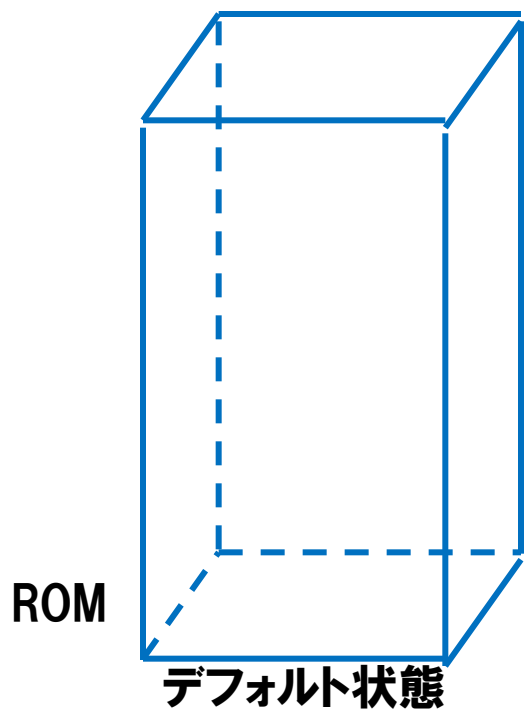
メモリ領域の変更(2/3)

■ メモリ領域の変更の例1(内蔵ROMの64Kバイトを、次の領域に分割する)

- 箱(メモリ領域)の一つ目: 名前ROM、開始位置(アドレス)0. 大きさ(サイズ)0x6000
- 箱(メモリ領域)の二つ目: 名前ROM2、開始位置(アドレス)0x6000. 大きさ(サイズ)0xa000

MEMORY ROM : (0H, 010000H)

MEMORY ROM : (0H, 006000H)
MEMORY ROM2 : (6000H, 00a000H)



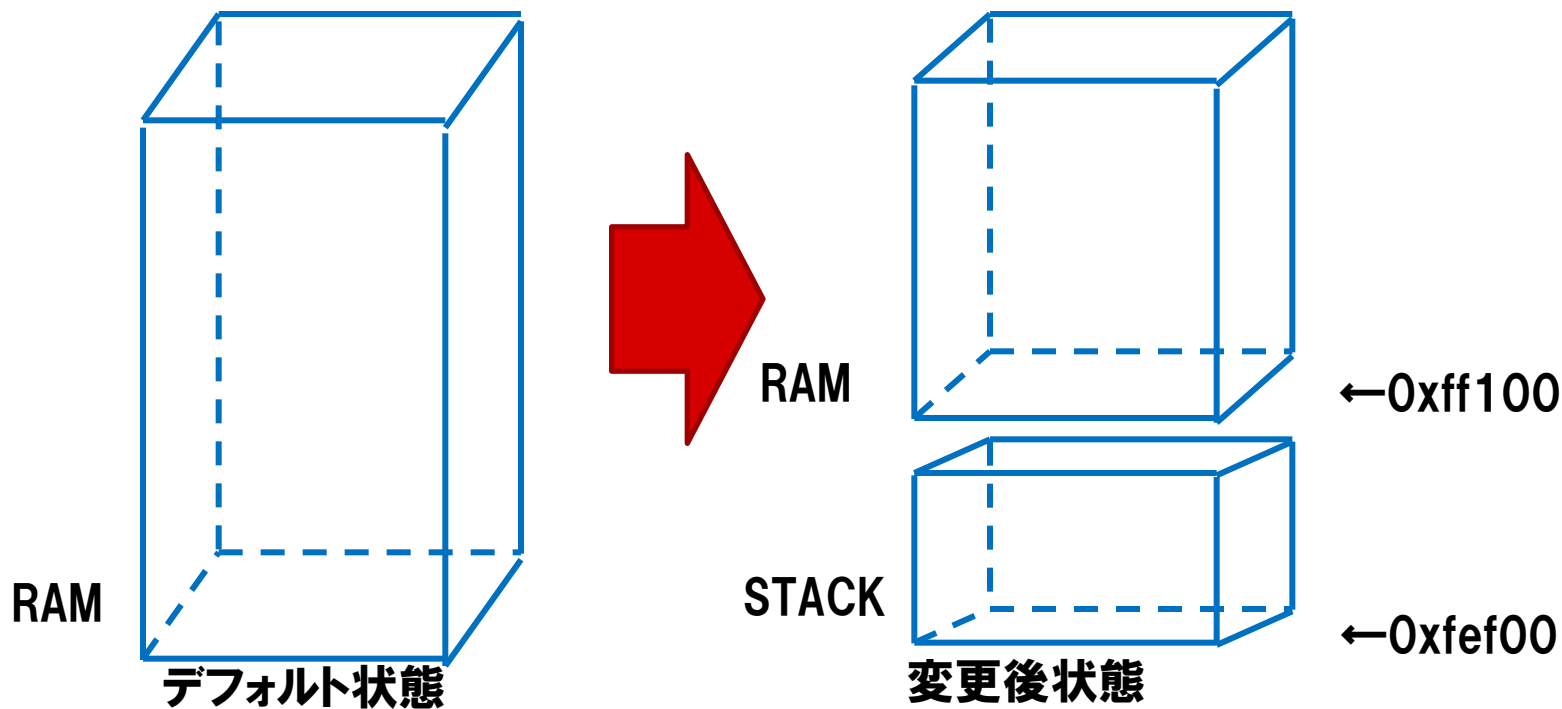
メモリ領域の変更(3/3)

■ メモリ領域の変更の例2(内蔵RAMを変数領域とスタック領域に分割する)

- 箱(メモリ領域)の一つ目: 名前STACK、開始位置(アドレス)0xfef00. 大きさ(サイズ)0x200
- 箱(メモリ領域)の二つ目: 名前RAM、開始位置(アドレス)0xff100. 大きさ(サイズ)0xe00

MEMORY RAM : (0FEF00H , 01100H)

MEMORY STACK : (0FEF00H, 0200H)
MEMORY RAM : (0FF100H, 0E00H)



セグメント配置の変更(1/3)

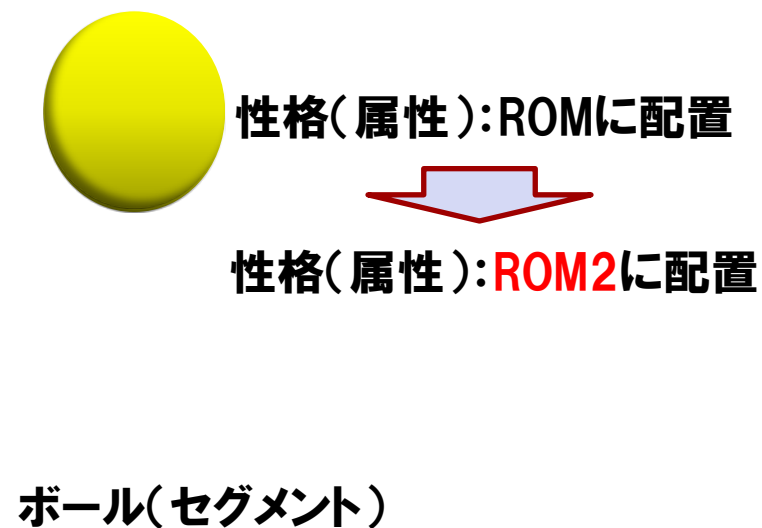
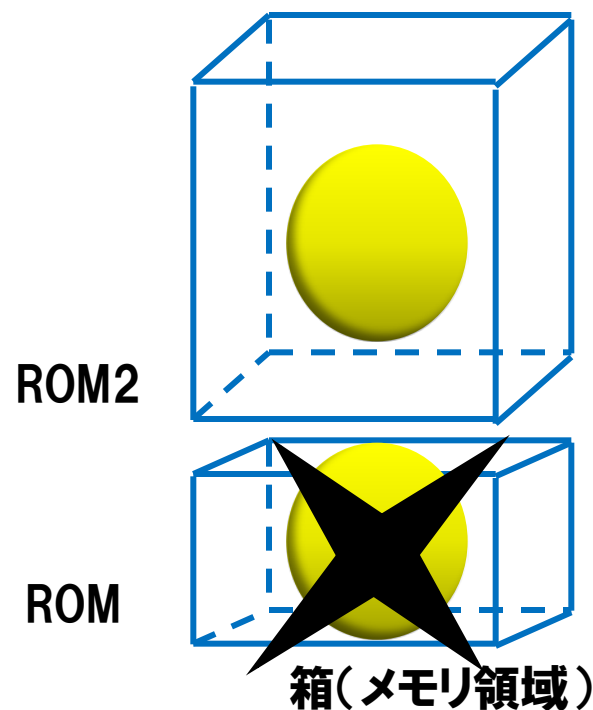
- ボール(セグメント)の位置(アドレス)を、デフォルトとは異なる性格(属性)で配置させる場合には、配置を指定する必要があります。
 - 配置する箱(メモリ領域)を変更する
 - 新たに配置したい箱(メモリ領域)を指定する
 - 配置する位置(アドレス)を変更する
 - 新たに配置したい位置(アドレス)を指定する

セグメント配置の変更(2/3)

- セグメント配置指定の例1(@@CODEセグメントをメモリ領域ROM2に配置する)

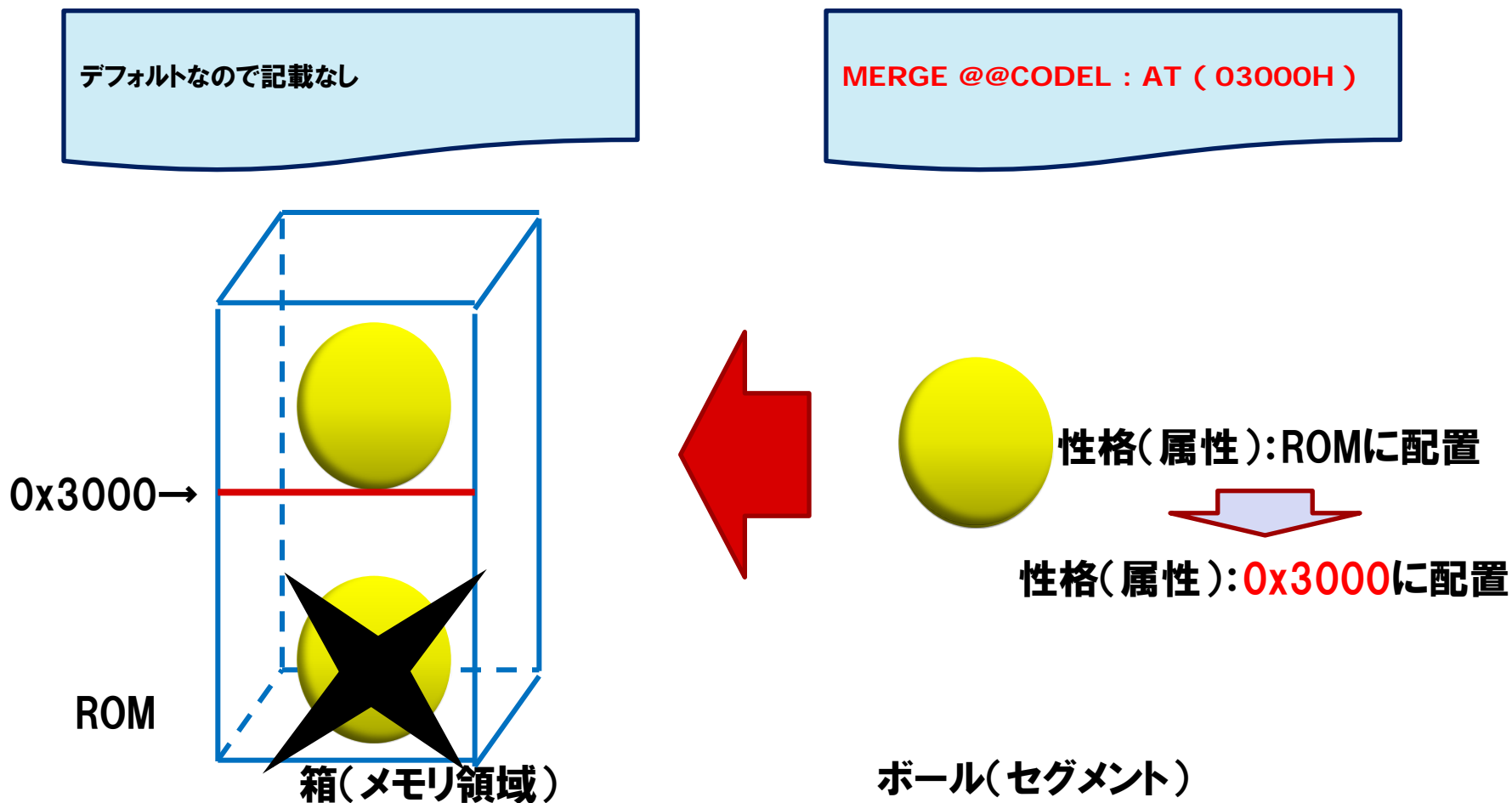
デフォルトなので記載なし

MERGE @@CODEL := ROM2



セグメント配置の変更(2/3)

- セグメント配置指定の例2(@@CODEセグメントを0x3000番地に配置する)



Cコンパイラが出力するセグメント名の変更

- Cコンパイラが出力するセグメントの配置を指定する場合に、分割して配置指定するには、セグメント名の変更が必要です。
- その場合には、CA78K0Rの `#pragma section` 指令を使用して、セグメント名の変更をして、そのセグメントをリンク・ディレクティブで指定してください。

例：Cソースでセグメント名(セクション名)の変更

```
#pragma section @@CODE CODE_SEG
```

例：リンク・ディレクティブでセグメントの配置を指定

```
MERGE CODE_SEG : AT ( 03000H )
```


Cコンパイラで出力されるセグメント名と属性(1/3)

■ CA78K0Rでデフォルトで出力されるセグメント(セクション)名と属性

セクション名	セグメント・タイプ	再配置属性	説明
@@CODE	CSEG	BASE	コード部用セグメント(near領域配置)
@@CODEL	CSEG		コード部用セグメント(far領域配置)
@@CODER	CSEG		RAM配置コード部用セグメント
@@LCODE	CSEG	BASE	ライブラリ・コード用セグメント(near領域配置)
@@LCODEL	CSEG		ライブラリ・コード用セグメント(far領域配置)
@@LCODER	CSEG		RAM配置ライブラリ・コード用セグメント
@@CNST	CSEG	MIRRORP	ROMデータ(near領域配置)
@@CNSTR	CSEG	MIRRORP (ミラー空間ありの場合) UNITP (ミラー空間なしの場合)	RAM配置ROMデータ用セグメント(near領域配置)
@@CNSTL	CSEG	PAGE64KP	ROMデータ(far領域配置)
@@CNSTLR	CSEG	PAGE64KP	RAM配置ROMデータ用セグメント(far領域配置)

Cコンパイラで出力されるセグメント名と属性(2/3)

■ CA78K0Rでデフォルトで出力されるセグメント(セクション)名と属性

セクション名	セグメント・タイプ	再配置属性	説明
@@R_INIT	CSEG	UNIT64KP	near初期化データ用セグメント(初期値あり)
@@RLINIT	CSEG	UNIT64KP	far初期化データ用セグメント(初期値あり)
@@R_INIS	CSEG	UNIT64KP	初期化データ用セグメント(初期値ありsreg変数)
@@CALT	CSEG	CALLT0	callt関数のテーブル用セグメント
@@VECTnn	CSEG	AT 00mmH	ベクタ・テーブル用セグメント
@@BASE	CSEG	BASE	callt関数・割り込み関数用セグメント
@@LBASE	CSEG	BASE	ライブラリ・callt関数用セグメント

Cコンパイラで出力されるセグメント名と属性(3/3)

■ CA78K0Rでデフォルトで出力されるセグメント(セクション)名と属性

セクション名	セグメント・タイプ	再配置属性	説明
@@INIT	DSEG	BASEP	データ領域用セグメント(初期値あり, near領域配置)
@@INITL	DSEG	UNIT64KP	データ領域用セグメント(初期値あり, far領域配置)
@@DATA	DSEG	BASEP	データ領域用セグメント(初期値なし, near領域配置)
@@DATAL	DSEG	UNIT64KP	データ領域用セグメント(初期値なし, far領域配置)
@@INIS	DSEG	SADDRP	データ領域用セグメント(初期値ありsreg変数)
@@DATS	DSEG	SADDRP	データ領域用セグメント(初期値なしsreg変数)
@@BITS	DSEG		boolean型変数, bit型変数用セグメント

高度なリンク・ディレクティブの機能の説明

- 前ページまでで説明していた機能に加えて、次の機能を持っています。
- **メモリ空間**
 - **箱(メモリ領域)**を横にならべて作成できます
 - これにより、**ボール(セグメント)**を同じアドレスに配置できます
 - デフォルトでは、**箱(メモリ領域)**は横に並べて作成されていません
- **結合属性**
 - 同じ名前の**ボール(セグメント)**をエラーにできます
 - これにより、1つしか作成しない**ボール(セグメント)**を複数作成したかのチェックが可能です
 - デフォルトでは、同じ名前の**ボール(セグメント)**は結合して1つにします

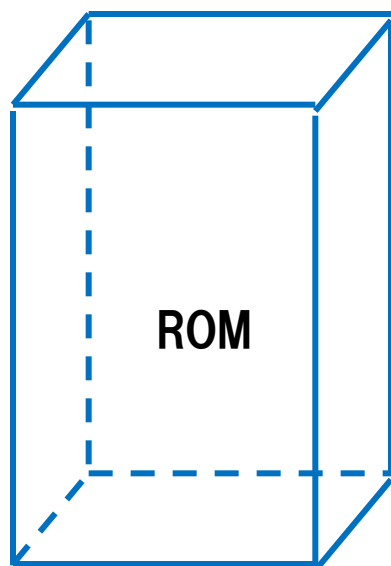
メモリ空間のイメージ

■ メモリ空間の例

- 箱(メモリ領域)の一つ目: 名前ROM、開始位置(アドレス)0. 大きさ(サイズ)0x10000
- 箱(メモリ領域)の二つ目: 名前ROM_EX1、開始位置(アドレス)0. 大きさ(サイズ)0x10000

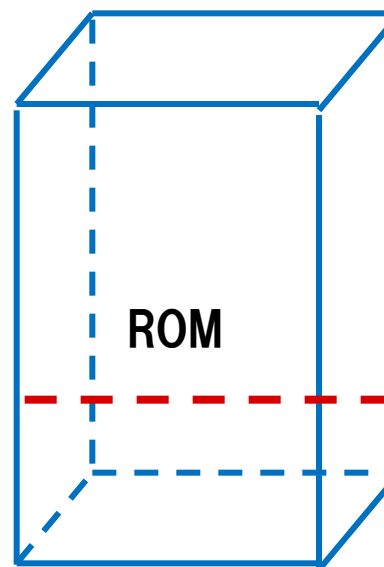
MEMORY ROM : (0H, 010000H)

MEMORY ROM : (0H, 010000H)
MEMORY ROM_EX1: (0H, 010000H)/EX1

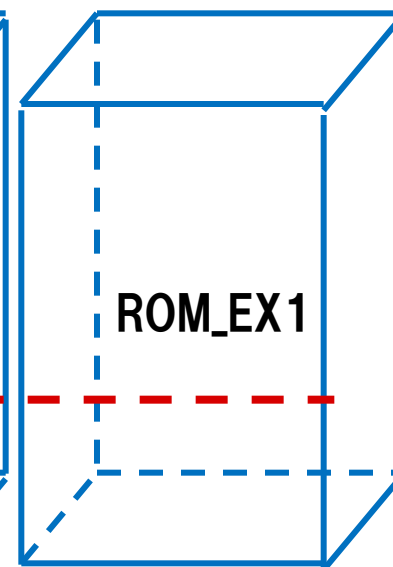


REGULAR

デフォルト状態



REGULAR



EX1

変更後状態

リンク・ディレクティブの記述形式(2)

■ メモリ・ディレクティブ

- MEMORY **メモリ領域名** : (**スタート・アドレス** , **サイズ**) [/ **メモリ空間名**]
 - **メモリ空間名**
 - 作成する**箱(メモリ領域)**の配置するメモリ空間を指定します
 - REGULAR, EX1, ..., EX15

■ セグメント配置ディレクティブ

- MERGE **セグメント名** : [AT (**スタート・アドレス**)] [= **メモリ領域名**] [/ **メモリ空間名**]
- MERGE **セグメント名** : [**結合属性**] [= **メモリ領域名**] [/ **メモリ空間名**]
 - **メモリ空間名**
 - 配置を指定する**ボール(セグメント)**のメモリ空間を指定します
 - REGULAR, EX1, ..., EX15
 - **結合属性**
 - 配置を指定する**ボール(セグメント)**の結合属性を指定します
 - SEQUENT(デフォルト)
セグメントを出現順に、順次空きを作らないようにマージします。BSEGはビット単位で出現順にマージします。
 - COMPLETE
同名のセグメントが複数存在する場合はエラーとします

注意: 前ページの説明では、よく使うもののみ説明していました。

CubeSuite+ 上でのリンク・ディレクティブ・ファイルの指定方法

■ リンク・ディレクティブ・ファイルをCubeSuite+ のプロジェクトに登録

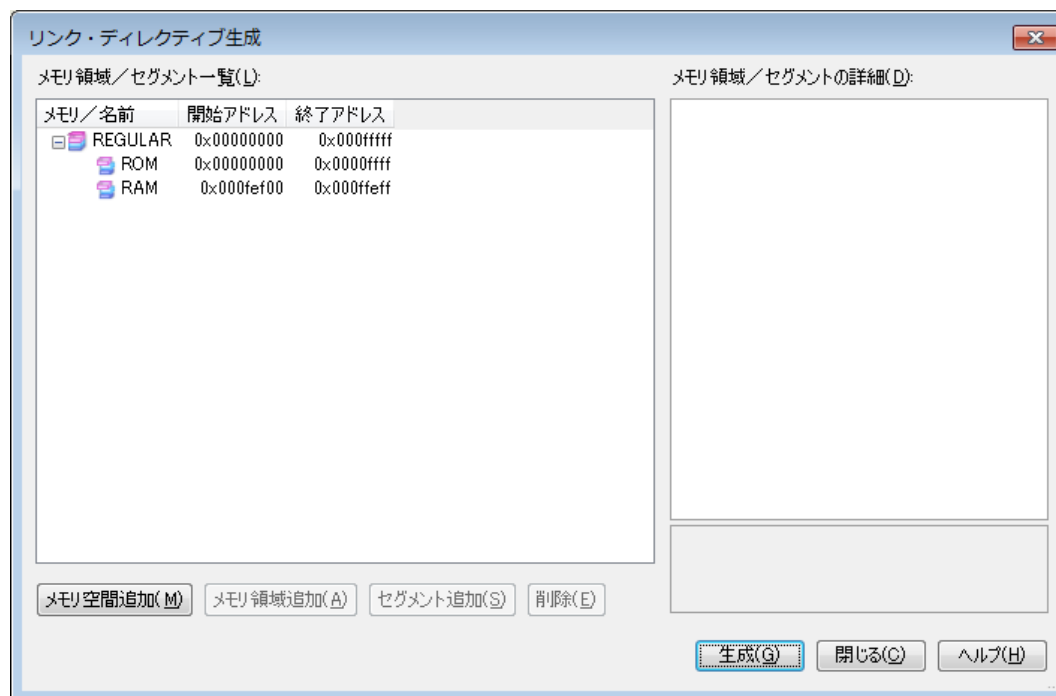
- CubeSuite+ のプロジェクトツリーパネルに、リンク・ディレクティブ・ファイルを、ドラッグ&ドロップすれば登録できます

The image illustrates the process of registering a link directive file into a CubeSuite+ project. On the left, a file explorer window shows a folder named '新しいフォルダー' containing files: 'DefaultBuild', 'LinkDirective.mtpj', 'main.c', and 'rl78.dr'. A red arrow points from 'rl78.dr' to the 'main.c' file in the project tree of the 'LinkDirective - CubeSuite+' window. The project tree shows a hierarchy: 'LinkDirective (プロジェクト)*' containing various tools and a 'ファイル' (Files) sub-folder. The 'main.c' file is highlighted in the 'ファイル' sub-folder. A yellow callout box with a red border contains the text: **登録されると リストに表示されます** (When registered, it will be displayed in the list).

```
int a;  
void main(void)  
{  
    a++;  
}
```

CubeSuite+ 上でのリンク・ディレクトィブの生成(1/2)

- CubeSuite+ の機能を使用して、リンク・ディレクトィブ・ファイルを生成することが可能です
 - プロジェクト・ツリー パネル の「CA78K0R(ビルド・ツール)」を右クリックし、「リンク・ディレクトィブ・ファイルを生成する」を選択



- 注意:リンク・ディレクトィブ・ファイルを読み込むことはできません

CubeSuite+ 上でのリンク・ディレクトティブの生成(2/2)

選択したメモリ領域、セグメントの詳細情報を表示
編集もここから可能！

The screenshot shows the 'Link Directory Generation' dialog box with two main panes. The left pane, titled 'メモリ領域/セグメント一覧(L):', contains a table of memory regions and segments. The right pane, titled 'メモリ領域/セグメントの詳細(D):', shows details for the selected 'ROM2' segment. A mouse cursor is shown dragging the 'ROM2' entry from the left pane to the right pane. At the bottom, there are buttons for 'メモリ空間追加(M)', 'メモリ領域追加(A)', 'セグメント追加(S)', '削除(E)', '生成(G)', '閉じる(C)', and 'ヘルプ(H)'. A small text box at the bottom right contains the text 'ここで使用できる文字は以下のとおり'.

メモリ/名前	開始アドレス	終了アドレス
REGULAR	0x00000000	0x000fffff
ROM	0x00000000	0x00005fff
ROM2	0x00006000	0x0000ffff
@@CODEL		-
RAM	0x000ff100	0x000ffeff
STACK	0x000fef00	0x000ff0ff

メモリ領域	
名前	ROM2
開始アドレス	HEX 6000
サイズ	HEX A000

リンク・ディレクトティブの内容を表示！

セグメントのメモリ領域配置指定の変更はドラッグ&ドロップで可能！

設定完了後に、ボタンを押すだけで
リンク・ディレクトティブ・ファイルを生成！
プロジェクトに自動登録！

メモリ空間、メモリ領域、セグメントの
生成はボタンを押すだけ！

RENEASAS

ルネサス ソリューションズ株式会社

© 2014 Renesas Solutions Corp.