

# RH850コンパイラ CC-RH 割り込み/例外

ルネサス システムデザイン株式会社  
ツールビジネス事業部 ツール技術部

2014/10/7 Rev. 1.00

R20UT3215JJ0100

# RH850の割り込み

RH850の割り込み/例外は、**直接ベクタ方式**と**テーブル参照方式**の2種類の方式があります。

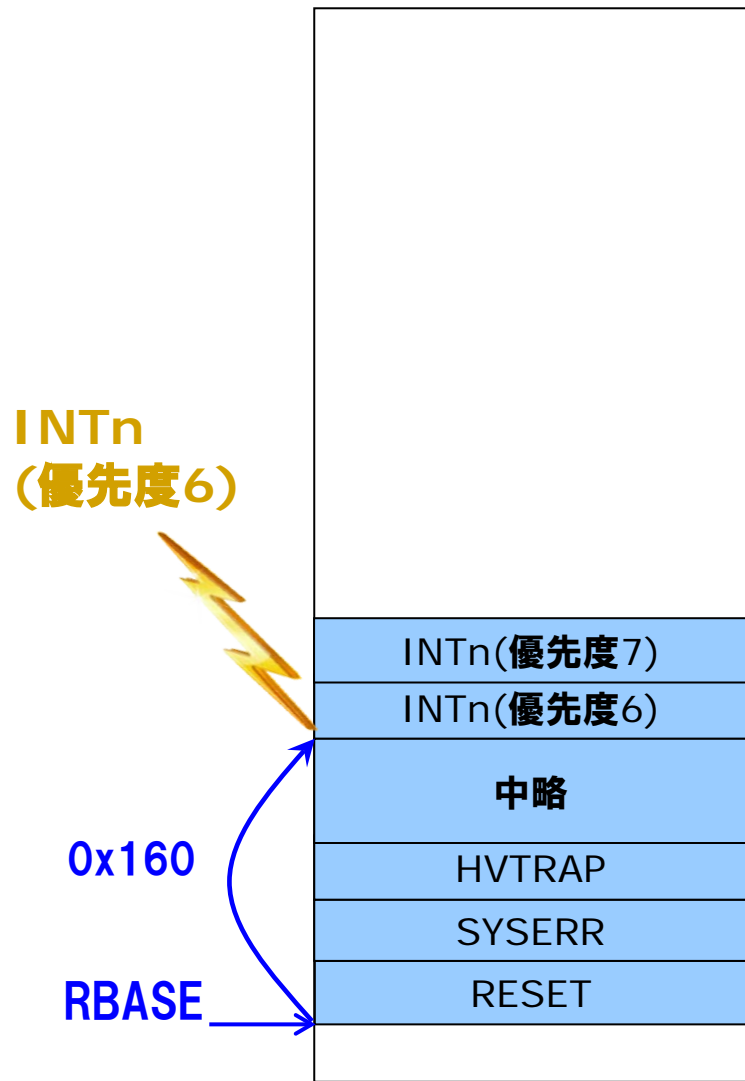
デフォルトでは直接ベクタ方式が選択されており、発生要因毎に固定のハンドラ・アドレスに分岐して、分岐先のコードを実行します。

例外/割り込み	PSW.EBV = 0	PSW.EBV = 1	RINT = 0	RINT = 1
	ベースレジスタ		オフセットアドレス	
RESET	RBASE	なし	000 <sub>H</sub>	000 <sub>H</sub>
SYSERR		EBASE	010 <sub>H</sub>	010 <sub>H</sub>
HVTRAP		020 <sub>H</sub>	020 <sub>H</sub>	
FETRAP		030 <sub>H</sub>	030 <sub>H</sub>	
TRAP0		040 <sub>H</sub>	040 <sub>H</sub>	
TRAP1		050 <sub>H</sub>	050 <sub>H</sub>	
RIE		060 <sub>H</sub>	060 <sub>H</sub>	
FPP/FPI		070 <sub>H</sub>	070 <sub>H</sub>	
UCPOP		080 <sub>H</sub>	080 <sub>H</sub>	
MIP/MDP/ITLBE/DTLBE		090 <sub>H</sub>	090 <sub>H</sub>	
PIE		0A0 <sub>H</sub>	0A0 <sub>H</sub>	

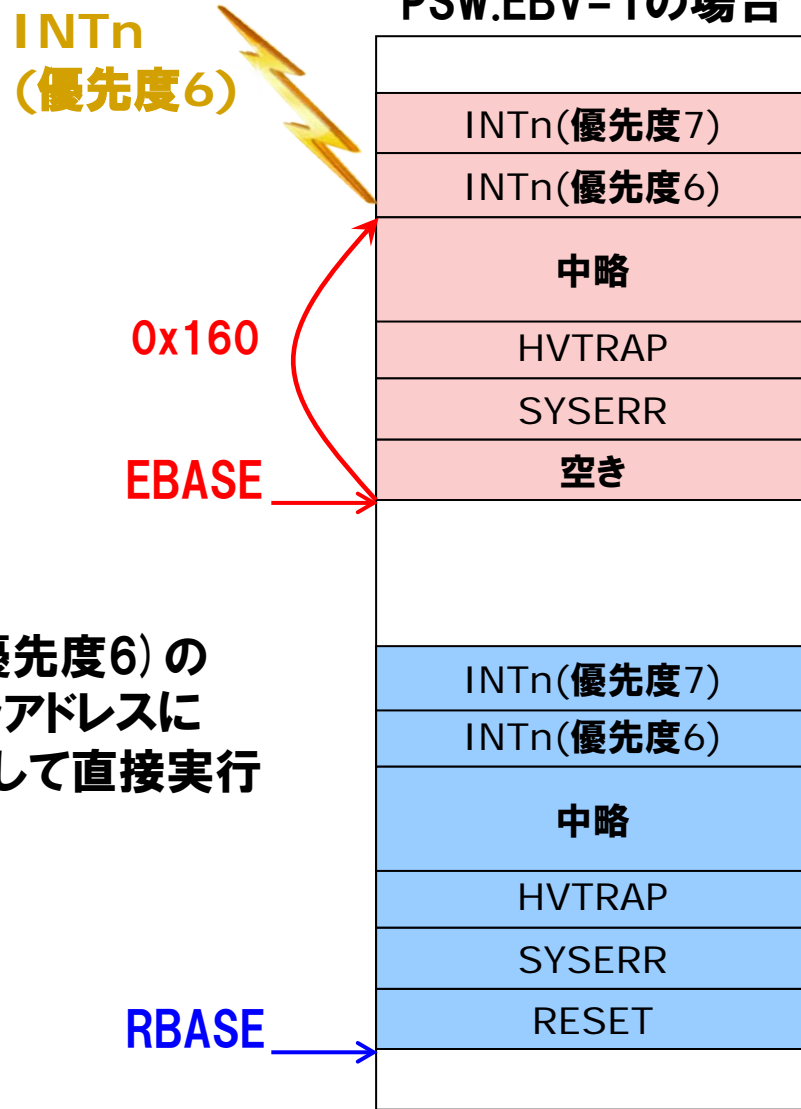
今回は直接ベクタ方式の割り込み/例外をCC-RHにより定義する方法を説明します。

# 直接ベクタ方式のイメージ

PSW.EBV=0の場合

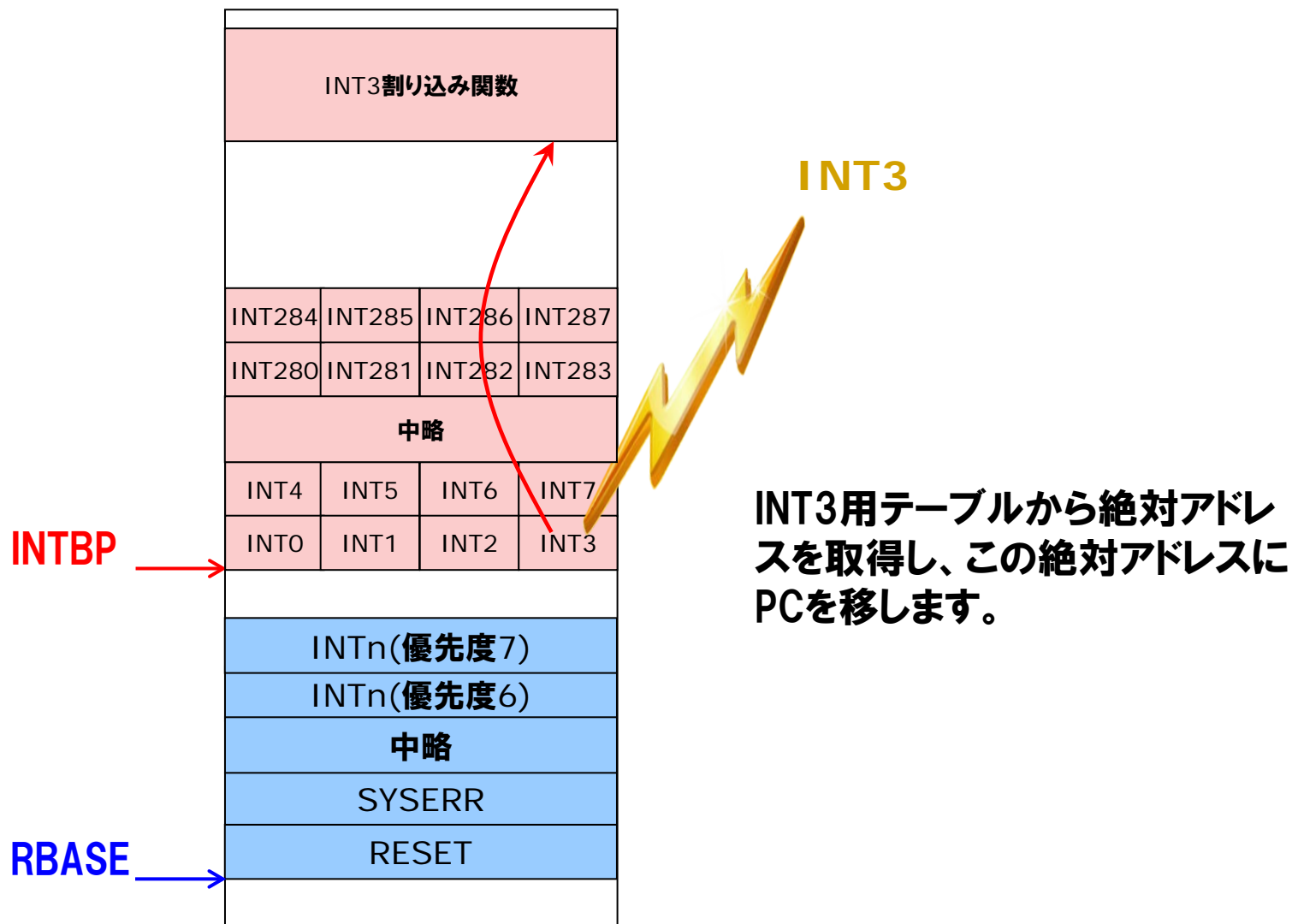


PSW.EBV=1の場合



INTn (優先度6) の  
ハンドラアドレスに  
PCを移して直接実行

# テーブル参照方式のイメージ



# 割り込み/例外ハンドラの記述方法

C言語で割り込み/例外関数を記述する場合、`#pragma interrupt` 指令を使用してください。

```
#pragma interrupt 関数名 (割り込み仕様 [, 割り込み仕様])
```

- 「**割り込み仕様**」に指定可能な文字列はマニュアルに記載しております。
- 「**関数名**」で指定した関数を割り込み/例外ハンドラとして、コード生成時に必要なレジスタの退避・復帰処理を追加します。

例:例外”SYSERR”発生時に割り込み関数”func”を実行する場合

```
#pragma interrupt func(priority=SYSERR, callt=true, fpu=true)
void func1(unsigned long feic)
{
    ユーザー記述処理;
}
```

# 割り込み/例外ハンドラの実行コードイメージ

```
_func:  
    movea -0x00000038, r3, r3  
    st23.dw r6, 0x00000030[r3]  
    stsr 0, r6  
    stsr 1, r7  
    st23.dw r6, 0x00000028[r3]  
    stsr 13, r6  
    ei  
    st23.dw r4, 0x00000020[r3]  
    st23.dw r8, 0x00000018[r3]  
    st23.dw r10, 0x00000010[r3]  
    stsr 16, r8  
    stsr 17, r9  
    st23.dw r8, 0x00000008[r3]  
    stsr 7, r8  
    stsr 6, r9  
    st23.dw r8, 0x00000000[r3]  
    prepare ....  
    : ; ユーザが記述した処理  
    dispose ....  
    syncce  
    ld23.dw 0x00000000[r3], r8  
    ldsr r9, 6  
    ldsr r8, 7  
    ld23.dw 0x00000008[r3], r8  
    ldsr r9, 17  
    ldsr r8, 16  
    ld23.dw 0x00000010[r3], r10  
    ld23.dw 0x00000018[r3], r8  
    ld23.dw 0x00000020[r3], r4  
    di  
    ld23.dw 0x00000028[r3], r6  
    ldsr r7, 1  
    ldsr r6, 0  
    ld23.dw 0x00000030[r3], r6  
    movea 0x00000038, r3, r3  
    eiret
```

※出力コードは#pragma interrupt の「割り込み仕様」で指定する文字列により異なります。

#pragma interrupt 指令により、割り込み関数先頭にコンパイラが埋め込む入口コード

#pragma interrupt 指令により、割り込み関数先頭にコンパイラが埋め込む出口コード

## 割り込み/例外ベクタ(1/2)

割り込み/例外ベクタとは、各ハンドラアドレスに割り込み/例外関数への分岐命令を記述したものです。直接ベクタ方式時に使用します。

CS+で新規プロジェクトを作成した場合には、割り込み/例外ベクタのサンプルとして”vecttbl.asm”がプロジェクトに登録されています。本ファイルを必要に応じてカスタマイズしてご使用ください。

```
.section "RESET", text
.align 512
jr32    __start ; RESET

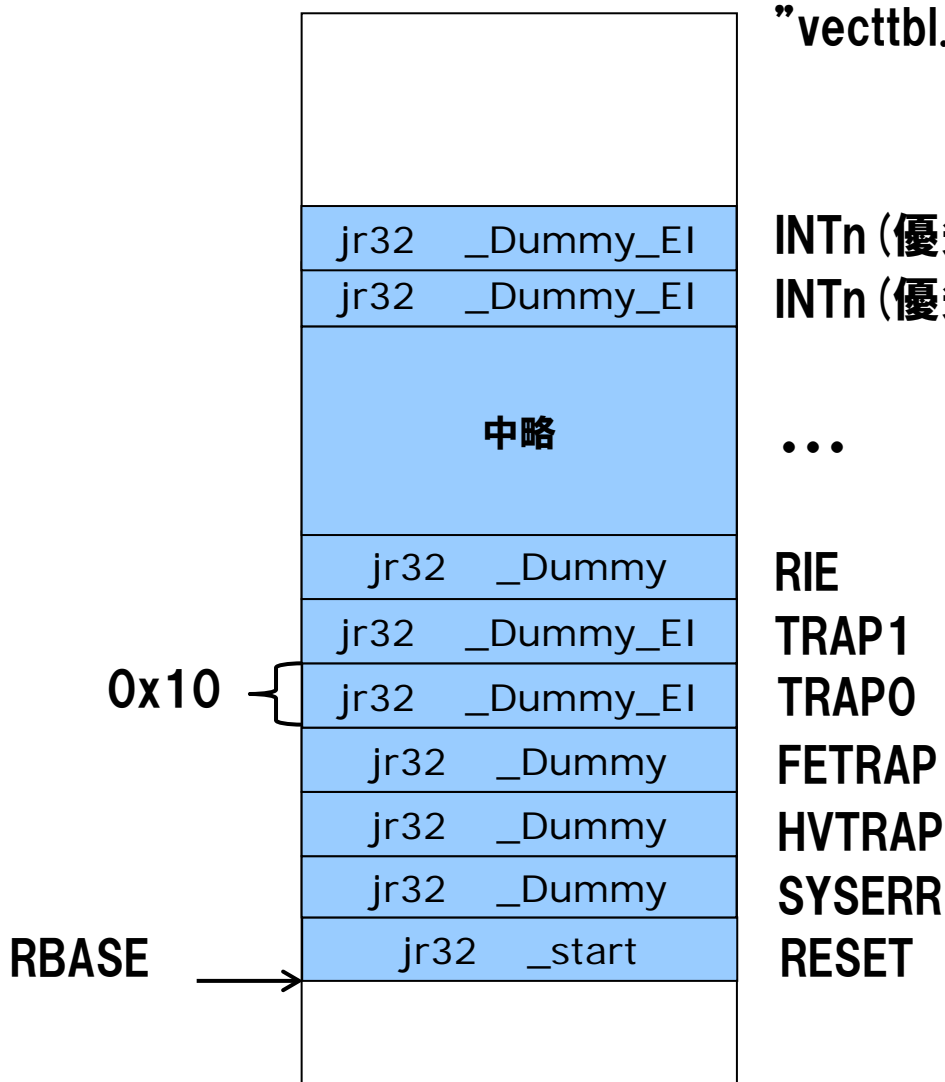
.align 16
jr32    _Dummy ; SYSERR

.align 16
jr32    _Dummy ; HVTRAP

...
```

デフォルトでは、割り込み/例外ベクタをRESETセクションとして、SYSERR/HVTRAP/FETRAP 等の対応するオフセット位置に、ダミー関数”\_Dummy”あるいは”Dummy\_EI”に分岐させる命令を配置しています。

## 割り込み/例外ベクタ(2/2)



“vecttbl.asm”のメモリ空間上のイメージ

INTn (優先度7)

INTn (優先度6)

...

RIE

TRAP1

TRAP0

FETRAP

HVTRAP

SYSERR

RESET

割り込み/例外ベクタはRESETセクションに配置されていますので、RESETセクションを適切なアドレスに配置させてください。

RBASEの初期値は0x00のため、初期値のまま使用するのであればRESETセクションを0x00番地に配置させてください。

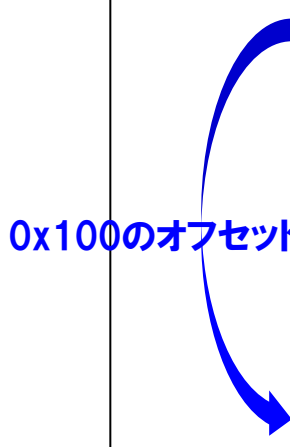


## 割り込み/例外の定義方法(1/4)

**各発生要因に対応するオフセットアドレスのジャンプ命令をお客様自身で定義した割り込み/例外ハンドラへのジャンプ命令に変更してください。**

**例: 割り込みINTn (優先度0) 発生時に割り込み関数”func”を実行する場合**

PSW.EBV=0の場合、INTn (優先度0) はRBASEレジスタ値から0x100のオフセットアドレスとなります。そのためRESETセクションの先頭から0x100の位置にあるジャンプ命令を変更してください。



```
.section "RESET", text
.align 512
jr32    __start ; RESET

.align 16
jr32    _Dummy ; SYSERR
...

.align 16
jr32    func ; INTn(priority0) ← 関数funcへの分岐命令に変更
...
```

## 割り込み/例外の定義方法(2/4)

割り込み/例外ベクタが配置されるRESETセクションを適切なアドレスに配置させてください。

例: 割り込みINTn (優先度0) がRBASEをベースレジスタとする場合

RESETセクションをRBASEレジスタ値と同じアドレスに配置させてください。RBASEレジスタ値の初期値0x00で使用する場合は、RESETセクションも0x00番地に配置させてください。



## 割り込み/例外の定義方法(3/4)

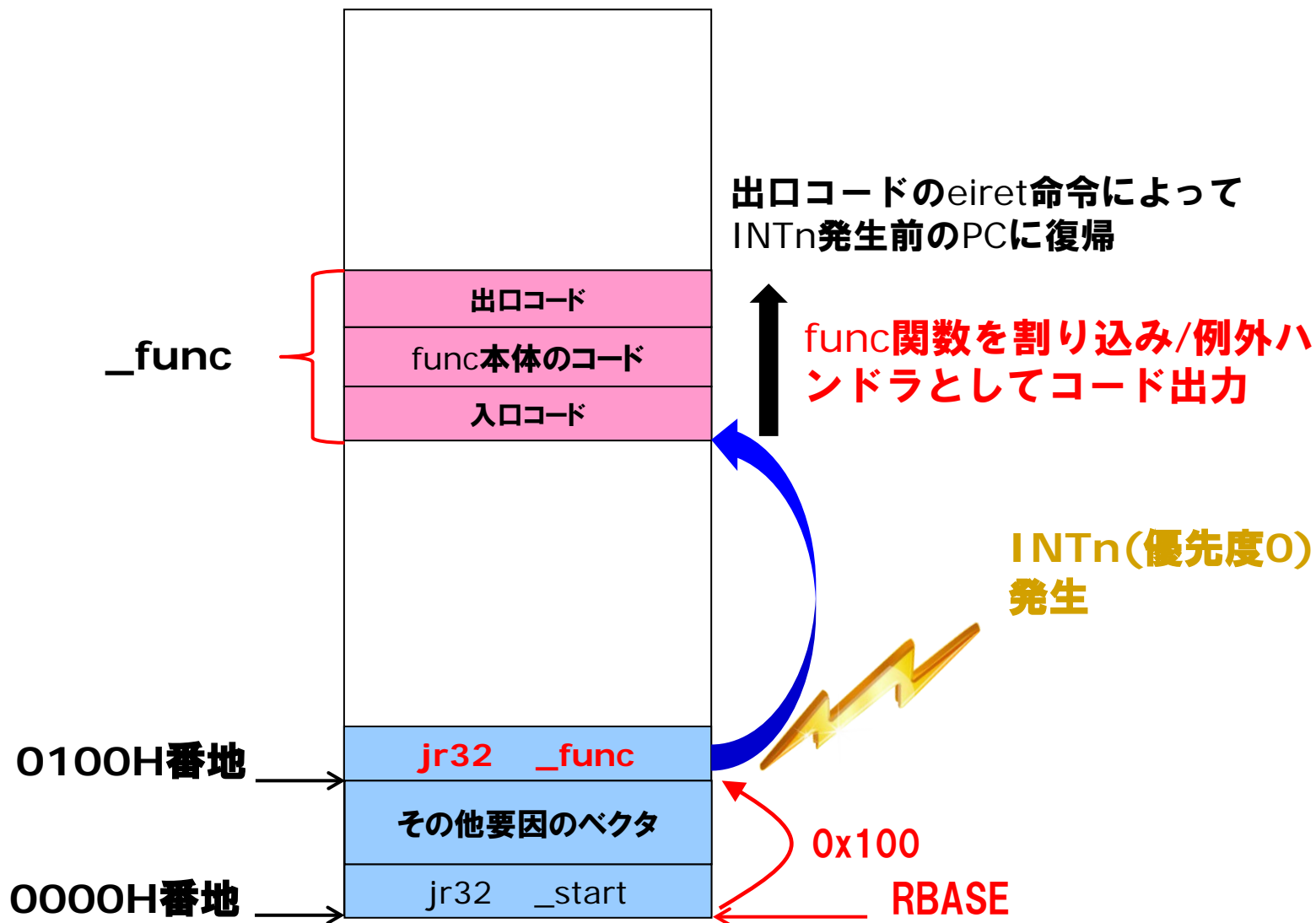
関数”func”を#pragma interrupt 指令を使用して割り込み/例外ハンドラとしてコンパイルしてください。

例:INTn (優先度0) の割り込み関数”func”を定義する場合

```
#pragma interrupt func(enable=true, callt=true, fpu=true)
void func(unsigned long eiic)
{
    ユーザ記述処理;
}
```

「割り込み仕様」に指定した規則に従い、関数”func”のコード生成時に入口コードと出口コードを生成します。

# 割り込み/例外の定義方法(4/4)





**ルネサス システムデザイン株式会社**

©2014 Renesas System Design Co., Ltd. All rights reserved.