SH7753 グループ

PCIe RC ドライバソフトウェア

R01AN1618JJ0103
Rev.1.03
2014.3.7

要旨

本仕様書はSH7753グループのRC（PCIe Root Complex）ドライバについて説明します。

対象デバイス

SH7753

動作環境

本仕様書に示す RC ドライバの動作環境を以下に示します。

- 評価ボード　SH7753 グループ EVB ボード
- ソフトウェア　: High-Performance Embedded Workshop- Ver 4.09.00.007
  : Toolchain - Ver 9.4.1.0
  : OptLinker - Ver 10.01.00
  : SH アセンブラ - Ver 7.01.02
  : SH C/C++コンパイラ- Ver 9.04.01
  : SH C/C++ライブラリジェネレータ - Ver 3.00.03
- エミュレータ　: E10A エミュレータ Ver 3.03.00

目次

1. 機能概要

本ドライバは SH7753 に搭載されている RC（Root Complex）機能を使用し、PCIe 接続された End Point デバイスにアクセスするための関数群で構成されています。

1.1 主な機能

RC ドライバの主な機能をを以下に示す。

- PCIe Link 層に対する RC 向け初期設定
- PCIe Link 層、Phy 層のステータス取得、機能設定
- Posted／Non-Posted／Completion トランザクションの送受信
- Memory トランザクションの DMA 転送
- RC 機能の割込み設定

1.2 関連ドキュメント

SH7753 グループ ユーザーズマニュアル：ハードウェア編

2. ドライバ仕様

2.1 関数一覧

表 1に、RC ドライバの関数一覧を示す。

表1. RC ドライバ関数一覧

分類	関数名	機能概要	スタック サイズ
リセット関数	R_RC_Reset	Root Complex リセット	4
初期設定関数	R_RC_InitReset	PCIe Link 初期設定（リセット解除時）	40
	R_RC_InitDLActive	PCIe Link 初期設定（DL Active 時）	64
Link 情報アクセス関数	R_RC_GetPhyStatus	PCIe Phy ステータス取得	4
	R_RC_GetLinkStatus	PCIe Link ステータス取得	8
	R_RC_GetRxStatus	受信ステータス取得	12
	R_RC_GetTxStatus	送信ステータス取得	4
	R_RC_GetErrPacket	エラーパケット情報取得	68
	R_RC_SetCplTimeout	コンプリーションタイムアウト設定	40
	R_RC_SetBusNum	バス／デバイス番号設定	40
	R_RC_BusReset	セカンダリバスリセット	36
	R_RC_SetExSync	拡張 Sync 設定処理	36
	R_RC_Retrain	Link リトレーニング	36
	R_RC_LinkDisable	Link Disable	36
	R_RC_SetASPM	ASPM サポートレベル設定	36
PM 制御関数	R_RC_SetPM	パワーマネージメント設定	48
	R_RC_SetL1	L1 遷移設定	40
TLP 送受信関数	R_RC_TxNP	Non-Posted 送信要求	96
	R_RC_TxP	Posted 送信要求	96
	R_RC_TxCpl	Completion 送信要求	96
	R_RC_RxNP	Non-Posted 受信要求	100
	R_RC_RxP	Posted 受信要求	104
	R_RC_RxCpl	Completion 受信要求	124
DMA 転送関数	R_RC_DMAStart	DMA 転送要求	32
	R_RC_DMAStop	DMA 転送停止	20
	R_RC_GetDMABusy	DMA ビジーステータス取得	4
	R_RC_GetDMACnt	DMA 転送カウントモニタ	0
割込み設定関数	R_RC_SetRCIntEn	RC-DMA 割り込み許可設定	48
	R_RC_GetRCIntEn	RC-DMA 割り込み許可設定取得	48
	R_RC_ClrRCInt	RC-DMA 割り込み要因クリア	60
	R_RC_GetRCInt	RC-DMA 割り込み要因取得	48
	R_RC_SetLinkIntEn	PCIe Link 割り込み許可設定	56
	R_RC_GetLinkIntEn	PCIe Link 割り込み許可設定取得	56
	R_RC_ClrLinkInt	PCIe Link 割り込み要因クリア	72
	R_RC_GetLinkInt	PCIe Link 割り込み要因取得	56
コールバック登録関数	R_RC_SetCallback	コールバック関数登録	0
割込みハンドラ関数	R_RC_Interrupt	PCIe 割込みハンドラ	4

2.2 コールバック関数仕様一覧

表 2に RC ドライバのコールバック関数仕様一覧を示す。

表2. RC ドライバコールバック関数仕様一覧

関数名	コールバック名	仕様内容
R_RC_SetCallback	void (*pv_rc_callback)(void)	PCIe 割込み発生時にコールされるコールバック。 （NULL の場合、コールバックなし）

2.3 エラーコード一覧

表 3に RC ドライバのエラーコード一覧を示す。

表3. RC ドライバエラーコード一覧

値	エラーコード（マクロ定義）	エラー内容
0	RET_NORMAL	正常終了
-1	RET_ERR_PARAM1	第 1 引数不正
-2	RET_ERR_PARAM2	第 2 引数不正
-20	RET_ERR_PM_L1	L0→L1 状態への遷移中により送信不可 送信にはリトライ実施のこと
-21	RET_ERR_L1_START	PM ステータスが L0 状態でないため、L1 遷移不可
-22	RET_ERR_LINK_ACCESS	LINK アクセスエラー

3. 関数

本章では、RC ドライバの各関数仕様詳細を示す。各関数詳細の読み方は以下のとおりです。

関数名 機能概要	分類
書式	関数の呼び出し形式を示します。#include “ヘッダファイル”で示すヘッダファイルは、この関数の実行に必要な標準ヘッダファイルで、必ずインクルードする。
引数	I,O は、引数がそれぞれ入力データ、出力データであることを意味する。
戻り値	関数の戻り値を示す。
解説	関数の仕様について説明する。
注意事項	注意事項があればここに示す。

R_RC_Reset

RC 設定関数

Root Complex リセット処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t  R_RC_Reset( uchar_t uc_rc_rst );
```

引数			
	uchar_t uc_rc_rst	I	RC リセット制御 (0：リセット解除、1：リセット)

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_PARAM1	第 1 引数不正

解説　本関数は、RC モジュールのリセット及び、リセット解除を行う。

注意　RC リセット時、他の関数をコールしないこと。
リセット期間は 20us 以上保持すること。

R_RC_InitReset

RC 設定関数

初期設定（リセット解除時）処理

書式

```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_InitReset( void );
```

引数

なし

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、RC リセット解除時の初期設定を行う。
Link デバイスに対して、Root Complex として動作するための初期設定を行い、トレーニングシーケンス開始までを実施する。

リセット解除時の使用例を6.1.1 リセット解除時の使用例に示す。

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_InitDLActive

RC 設定関数

初期設定（DL Active 時）処理

書式

```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_InitDLActive( void );
```

引数　なし

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、DL Active 時の初期設定を行う。
Link デバイスに対して、送受信バスの起動処理を実施する。

DL Active 時の使用例を6.1.2 DL Active 時の使用例に示す。

注意　本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetPhyStatus

RC 設定関数

PCIe Phy ステータス取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_GetPhyStatus( void );
```

引数　なし

戻り値		
0～1	PCIe Phy ステータス （1：PCIe Phy 動作中、0：PCIe Phy 停止中）	

解説　本関数は、PCIe Phy ステータスの情報取得を行う。

注意　PCIe Phy 停止中は、以下の関数以外はコールを行わないこと。

- R_RC_GetPhyStatus
- R_RC_Reset
- R_RC_SetRCIntEn
- R_RC_GetRCIntEn
- R_RC_ClrRCInt
- R_RC_GetRCInt

R_RC_GetLinkStatus

RC 設定関数

Link ステータス取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
void R_RC_GetLinkStatus( rc_link_stat_t *pst_status );
```

引数

rc_link_stat_t *pst_status	0	Link ステータス情報構造体へのポインタ

戻り値

なし

解説

本関数は、Link ステータスの情報取得を行う。

【Link ステータス情報構造体】

ushort_t us_status0
- Bit14：（0：L1 遷移は完了・中断していない、1：L1 遷移は完了・中断）
- Bit12：（0：PM_Enter_L1 DLLP 未受信、1：PM_Enter_L1 DLLP 受信）
- Bit11-9：（000：LDn、001：L0、011：L1、010：L2、100：L0s）
- Bit8：（0：DL Active 状態でない、1：DL Active 状態）
- Bit7：（0：DL Down 状態でない、1：DL Down 状態）
- Bit6：（0：DLLP Error 発生していない、1：DLLP Error 発生）
- Bit5：（0：Replay TimeOut 発生していない、1：Replay TimeOut 発生）
- Bit4：（0：Replay Number Roll Over 発生していない、1：Replay Number Roll Over 発生）
- Bit3：（0：BAD TLP 検出していない、1：BAD TLP 検出）
- Bit2：（0：BAD DLLP 検出していない、1：BAD DLLP 検出）

ushort_t us_status1
- Bit11：（0：Receiver Error 検出していない、1：Receiver Error 検出）
- Bit10：（0：Link Training 中でない、1：Link Training 中）

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetRxStatus

RC 設定関数

受信ステータス取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
void R_RC_GetRxStatus( rc_rx_stat_t *pst_status );
```

引数			
rc_rx_stat_t *pst_status	0	受信ステータス情報構造体へのポインタ

戻り値　なし

解説　本関数は、受信ステータスの情報取得を行う。

【受信ステータス情報構造体】

ushort_t us_status0
- Bit9：Completion タイムアウトステータス（1：タイムアウト発生）
- Bit8：Posted エラーステータス（1：受信バッファエラー）
- Bit7：NonPosted エラーステータス（1：受信バッファエラー）
- Bit6：Completion エラーステータス（1：受信バッファエラー）
- Bit5：Posted フルステータス（1：受信バッファフル）
- Bit4：NonPosted フルステータス（1：受信バッファフル）
- Bit3：Completion フルステータス（1：受信バッファフル）

uchar_t uc_status1
- Bit7-0：Completion タグ（NonPosted 送信時のヘッダタグ情報）

uchar_t uc_status2
- Bit6-0：Completion ヘッダ数（0～2：マルチコンプリーション受信時のヘッダ数）

注意　本関数は、PCIe Phy 停止中にコールしないこと。
受信可能なコンプリーションデータサイズは最大で 32DW のため、受信する Completion ヘッダ数は最大で 2 個となる。

R_RC_GetTxStatus

RC 設定関数

送信ステータス取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
void R_RC_GetTxStatus( rc_tx_stat_t *pst_status );
```

引数			
rc_tx_stat_t *pst_status	0	送信ステータス情報構造体へのポインタ

戻り値　なし

解説　本関数は、送信ステータスの情報取得を行う。

【送信ステータス情報構造体】

ushort_t us_status0

Bit2：Posted 送信バッファエンプティ（1：送信バッファエンプティ）

Bit1：NonPosted 送信バッファエンプティ（1：送信バッファエンプティ）

Bit0：Completion 送信バッファエンプティ（1：送信バッファエンプティ）

注意　本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetErrPacket

RC 設定関数

エラーパケット情報取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_GetErrPacket( rc_tlp_h_t *pun_tlp_h, rc_err_packet_t *pun_err );
```

引数

rc_tlp_h_t *pun_tlp_h	O	エラーパケットヘッダデータ共用体へのポインタ
rc_err_packet_t *pun_err	O	エラー要因情報共用体へのポインタ

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、エラーパケット情報の取得を行う。

エラーパケットバッファが Full 状態であれば、本関数をコールすることでエラーパケットヘッダとエラー要因を取得する。

【エラーパケットヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

【エラー要因情報共用体】

```
uchar_t uc_err
struct {
    uchar_t mft:1       (1 : Malformed TLP)
    uchar_t pst:1       (1 : Poisoned TLP)
    uchar_t ur:1        (1 : Unsupported Request)
    uchar_t ucpl:1      (1 : Unexpected Completion)
    uchar_t povf:1      (1 : Posted OverFlow)
    uchar_t npovf:1     (1 : Non Posted OverFlow)
    uchar_t ecrcf:1     (1 : ECRC Failed)
    uchar_t :1            Reserverd
} b;
```

注意

本関数をコールすることで、エラーパケットバッファ Full はクリアする。

本関数はエラーパケットバッファ Full 条件が成立時のみコールすること。エラーパケットバッファ Full 条件は以下で判断できる。

- Link 割込みのエラーパケット受信割込みまたは、割込み要因。
- RC-DMA 割込みのエラーパケット受信割込みまたは、割込み要因。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetCplTimeout

RC 設定関数

コンプリーションタイムアウト時間設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetCplTimeout( uchar_t uc_time );
```

引数			
	uchar_t uc_time	I	コンプリーションタイムアウト時間 （0x0A～0x32：10～50ms、0：タイムアウト無効）

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、コンプリーションタイムアウト時間の設定を行う。
10～50ms 間でタイムアウト時間の設定が可能である。0ms に設定した場合は、コンプリーションタイムアウト機能を無効とする。

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetBusNum

RC 設定関数

バス／デバイス番号設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetBusNum( uchar_t uc_bus, uchar_t uc_dev );
```

引数			
	uchar_t uc_bus	I	バス番号 (0～255)
	uchar_t uc_dev	I	デバイス番号 (0～31)

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM2	第 2 引数不正
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、リクエスト ID として使用するバス番号とデバイス番号の設定を行う。
Link デバイスは、自身の発行したリクエストに対するコンプリーション受信であるかを本関数で設定したバス／デバイス番号よりチェックする。

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_BusReset

RC 設定関数

セカンダリバスリセット処理

書式
```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_BusReset( uchar_t uc_bus_rst );
```

引数			
	uchar_t uc_bus_rst	I	セカンダリバスリセット設定 （1：リセット、0：リセット解除）

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_PARAM1	第 1 引数不正
	RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説　本関数は、セカンダリ I/F に対して Hot Reset 要求を実施する。

注意　本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetExSync

RC 設定関数

拡張 Sync 設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetExSync( uchar_t uc_sync );
```

引数			
	uchar_t uc_sync	I	拡張 Sync 設定 （1：許可、0：禁止）

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説　本関数は、MAC LTSSM が LOs からの遷移または、Recovery 状態への遷移時に、追加のオーダードセットを送信する。

注意　本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_Retrain

RC 設定関数

Link リトレーニング処理

書式
```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_Retrain( uchar_t uc_retrain );
```

引数			
	uchar_t uc_retrain	I	Link リトレーニング設定 （1：Link リトレーニング開始）

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説 本関数は、Link リトレーニングを指示することで、再度 Link トレーニングを実施する。

注意 本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_LinkDisable

RC 設定関数

Link Disable 処理

書式	#include “r_common.h” #include “iodefine.h” #include “r_rc_if.h” #include “rc_driver.h “ char_t R_RC_Linkdisable(uchar_t uc_disable);		
引数	uchar_t uc_disable	I	Link Disable 設定 （1：許可、0：禁止）
戻り値	RET_NORMAL		正常終了
	RET_ERR_PARAM1		第 1 引数不正
	RET_ERR_LINK_ACCESS		LINK アクセスエラー
解説	本関数は、Link を Disable にする。		
注意	本関数は、PCIe Phy 停止中にコールしないこと。		

R_RC_SetASPM

RC 設定関数

ASPM サポートレベル設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetASPM( uchar_t uc_aspm );
```

引数			
	uchar_t uc_aspm	I	ASPM サポートレベル （1：許可、0：禁止）

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説　本関数にて、ASPM のサポートレベル（L0s への自動遷移）を設定する。

注意　本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetPM

RC 設定関数

パワーマネージメント設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetPM( uchar_t uc_status );
```

引数			
	uchar_t uc_status	I	デバイスパワーステータス設定 （1：D3Hot、0：D0）

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_PARAM1	第 1 引数不正
	RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Root Complex デバイスのパワーステータス設定を行う。
D3Hot 指定時は PM 割込みを許可し、D0 指定時は禁止に戻す。

パワーマネージメント設定例を6.1.8　PM 遷移/復帰時の使用例に示す。

注意

本関数は、PCIe Phy 停止中にコールしないこと。
本関数によるパワーマネージメント制御時は PM 割込みを使用しているため、PM 割込みの設定は変更しないこと。

R_RC_SetL1

RC 設定関数

L1 遷移設定

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetL1( void );
```

引数　なし

戻り値

RET_NORMAL	正常終了
RET_ERR_L1_START	PM ステータスが L0 状態でないため、L1 遷移不可
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説　本関数は、Root Complex デバイスの L1 遷移を開始する。

パワーマネージメント設定例を6.1.8　PM 遷移/復帰時の使用例に示す。

注意　本関数は、PCIe Phy 停止中にコールしないこと。
本関数によるパワーマネージメント制御時は PM 割込みを使用しているため、PM 割込みの設定は変更しないこと。

R_RC_TxNP

RC 設定関数

NonPosted 送信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_TxNP( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data );
```

引数

rc_tlp_h_t *pun_tlp_h	I	送信ヘッダデータ共用体へのポインタ
ulong_t *pul_data	I	送信データへのポインタ

戻り値

RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_PM_L1	L0→L1 状態への遷移中により送信不可
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、NonPosted トランザクションの送信要求を行う。

送信ヘッダデータはトランザクションに応じたフォーマットで作成すること。送信ヘッダデータのトランザクションタイプが NonPosted でない場合は、RET_ERR_PARAM1（引数不正）を返す。その他の送信ヘッダ構成要素についてはチェックしない。

リクエストに対するコンプリーション受信は、Conpletion 受信要求関数にて行う。

NonPosted 送信要求時の使用例を6.1.3 NonPosted 送信時の使用例に示す。

【送信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

本関数は、PCIe Phy 停止中にコールしないこと。

受信可能なコンプリーションデータサイズは最大で 32DW までのため、32DW を超えるメモリーリード要求は行わないこと。

R_RC_TxP

RC 設定関数

Posted 送信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_TxP( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data );
```

引数

rc_tlp_h_t *pun_tlp_h	I	送信ヘッダデータ共用体へのポインタ
ulong_t *pul_data	I	送信データへのポインタ

戻り値

RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_PM_L1	L0→L1 状態への遷移中により送信不可
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Posted トランザクションの送信要求を行う。

送信ヘッダデータはトランザクションに応じたフォーマットで作成すること。送信ヘッダデータのトランザクションタイプが Posted でない場合は、RET_ERR_PARAM1（引数不正）を返す。その他の送信ヘッダ構成要素についてはチェックしない。

Posted 送信要求時の使用例を6.1.4 Posted 送信時の使用例に示す。

【送信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_TxCpl

RC 設定関数

Completion 送信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_TxCpl( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data );
```

引数			
	rc_tlp_h_t *pun_tlp_h	I	送信ヘッダデータ共用体へのポインタ
	ulong_t *pul_data	I	送信データへのポインタ

戻り値	
RET_NORMAL	正常終了
RET_ERR_PARAM1	第 1 引数不正
RET_ERR_PM_L1	L0→L1 状態への遷移中により送信不可
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Completion トランザクションの送信要求を行う。

送信ヘッダデータはトランザクションに応じたフォーマットで作成すること。送信ヘッダデータのトランザクションタイプが Completion でない場合は、RET_ERR_PARAM1（引数不正）を返す。その他の送信ヘッダ構成要素についてはチェックしない。

Completion 送信要求時の使用例を6.1.5 NonPosted 受信時の使用例に示す。

【送信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_RxNP

RC 設定関数

NonPosted 受信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_RxNP( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data, uchar_t *puc_err );
```

引数			
rc_tlp_h_t *pun_tlp_h	0	受信ヘッダデータ共用体へのポインタ
ulong_t *pul_data	0	受信データへのポインタ
uchar_t *puc_err	0	エラーステータスへのポインタ（1：エラー発生） Bit7：Malformed TLP Bit6：Poisoned TLP Bit5：Unsupported Request Bit1：ECRC Failed

戻り値	
RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、NonPosted トランザクションの受信要求を行う。
本関数をコールすることで受信ヘッダ、受信データ、受信ヘッダエラーステータスを取得する。
ただし、Malformed 及び、Unsupported Request 時は受信データを取得しない。

NonPosted 受信要求時の使用例を6.1.5 NonPosted 受信時の使用例に示す。

【受信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

受信データの格納場所は 1DW（NonPosted Write Data Size）確保すること。
本関数をコールすることで、NonPosted バッファ Full はクリアする。
本関数は NonPosted バッファ Full 条件が成立時のみコールすること。NonPosted バッファ Full 条件は以下で判断できる。

- Link 割込みの NonPosted 受信割込みまたは、割込み要因。
- 受信ステータス情報（R_RC_GetRxStatus）の NonPosted 受信バッファフルステータス。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_RxP

RC 設定関数

Posted 受信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_RxP( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data, uchar_t *puc_err );
```

引数			
rc_tlp_h_t *pun_tlp_h	O	受信ヘッダデータ共用体へのポインタ
ulong_t *pul_data	O	受信データへのポインタ
uchar_t *puc_err	O	エラーステータスへのポインタ（1：エラー発生） Bit7：Malformed TLP Bit6：Poisoned TLP Bit5：Unsupported Request Bit1：ECRC Failed

戻り値	
RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Posted トランザクションの受信要求を行う。
本関数をコールすることで受信ヘッダ、受信データ、受信ヘッダエラーステータスを取得する。
ただし、Malformed 及び、Unsupported Request 時は受信データを取得しない。

Posted 受信要求時の使用例を6.1.6 Posted 受信時の使用例に示す。

【受信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

受信データの格納場所は 32DW（Max Payload Size）確保すること。
本関数をコールすることで、Posted バッファ Full はクリアする。
本関数は Posted バッファ Full 条件が成立時のみコールすること。Posted バッファ Full 条件は以下で判断できる。

- Link 割込みの Posted 受信割込みまたは、割込み要因。
- 受信ステータス情報（R_RC_GetRxStatus）の Posted 受信バッファフルステータス。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_RxCpl

RC 設定関数

Completion 受信要求処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_RxCpl( rc_tlp_h_t *pun_tlp_h, ulong_t *pul_data, uchar_t *puc_err, uchar_t *puc_hnum );
```

引数

rc_tlp_h_t *pun_tlp_h	O	受信ヘッダデータ共用体へのポインタ	
ulong_t *pul_data	O	受信データへのポインタ	
uchar_t *puc_err	O	エラーステータスへのポインタ（1：エラー発生）	
		Bit7：Malformed TLP	
		Bit6：Poisoned TLP	
		Bit1：ECRC Failed	
		Bit0：Completion Time Out	
uchar_t *puc_hnum	O	Multiple Completion 時の受信ヘッダ数	

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Completion トランザクションの受信要求を行う。
本関数をコールすることで受信ヘッダ、受信データ、受信ヘッダエラーステータスを取得する。
Multiple Completion 受信時は、受信ヘッダ数分の Completion を取得する。
ただし、Completion TimeOut 時は Completion 取得を行わない。また、Malformed 時は受信データを取得しない。
Multiple Completion 時は受信ヘッダ数分のヘッダ情報を返すため、受信ヘッダデータ共用体ポインタ（*pun_tlp_h）には必要な格納領域を確保しておく。

Completion 受信要求時の使用例を6.1.3 NonPosted 送信時の使用例に示す。

【受信ヘッダデータ共用体】

```
ulong_t  ul_header[4];
ushort_t us_header[8];
uchar_t  uc_header[16];
```

注意

受信データの格納場所は NonPosted で要求する受信データ長以上確保すること。
本関数をコールすることで、Completion バッファ Full はクリアする。
本関数は Completion バッファ Full 条件が成立時のみコールすること。Completion バッファ Full 条件は以下で判断できる。

- Link 割込みの Completion 受信割込みまたは、割込み要因。
- 受信ステータス情報（R_RC_GetRxStatus）の Completion 受信バッファフルステータス。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_DMAStart

RC 設定関数

DMA 転送開始処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_DMAStart( rc_dma_ctrl_t *pst_dma );
```

引数			
	rc_dma_ctrl_t *pst_dma	I	DMA 転送管理構造体へのポインタ

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_PARAM1	第 1 引数不正

解説

本関数は、DMA 転送による Memory Read/Write トランザクションの発行を行う。

最大 64MB までの転送が可能である。よって、1 トランザクションの転送サイズ（ul_rcmemcfg b6-4）が 32DW の場合は、送信回数指定（ul_rctlptccfgr b23-0）に可能な設定値は最大で 524,288 回までとなる。

【DMA 転送管理構造体】

ulong_t ul_rcmemcfg
- Bit8：送信・受信データのバイトエンディアン設定（0：Big Endian、1：Little Endian）
- Bit6-4：1 トランザクションの転送サイズ（000：1DW、001：2DW、010：4DW、011：8DW、100：16DW、101：32DW）
- Bit3：MemWr 時の TLP 受信判定（0：無効、1：TLP 受信でエラー扱いとする）
- Bit2：TAG 選択（0：5bit TAG、1：8bit TAG）
- Bit1：Memory 転送方向（0：MemWr、1：MemRd）
- Bit0：Memory 転送アドレッシング選択（0：32bit、1：64bit）

ulong_t ul_rctlptccfgr
- Bit23-0：Memory トランザクション送信回数指定（H’00_0001～FF_FFFF：1～16,777,215 回、H’00_0000：16,777,216 回）

ulong_t ul_rcmemhdr[4]
- Memory トランザクションヘッダ（※ヘッダフォーマットを0 DMA 転送ヘッダ構成に示す）

ulong_t ul_ddr_addr
- DDR アドレス（MemWr 時：転送元、MemRd 時：転送先）

uchar_t uc_dmac_ch
- 汎用 DMAC の使用チャネル（0x06～0x0B：チャネル 6～11）

DMA 転送要求時の使用例を6.1.7 DMA 転送時の使用例に示す。

注意

この関数を使用する場合、汎用 DMAC（チャネル 6～11 の何れか一つ）を確保すること。

本関数は NonPosted/Posted/Completion 受信バッファ Full を全てクリアしてからコールすること。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_DMAStop

RC 設定関数

DMA 転送停止処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_DMAStop( uchar_t uc_dmac_ch );
```

引数			
	uchar_t uc_dmac_ch	I	汎用 DMAC の使用チャネル （0x06～0x0B：チャネル 6～11）

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_PARAM1	第 1 引数不正

解説

本関数は、DMA 転送による Memory Read/Write トランザクションの停止を行う。
本関数をコールすることにより、実行中の DMA 転送を終了する。

注意

DMA 転送中断前に発行した Memory Read のコンプリーション応答が中断後に発生する可能性がある。
Memory Read の DMA 転送中断時は、中断後にコンプリーションタイムアウト時間以上の間隔を空けてから、コンプリーションの受信 Full クリア（R_RC_RxCpl）を実施すること。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetDMABusy

RC 設定関数

DMA ビジーステータス取得処理

書式
```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_GetDMABusy( void );
```

引数 なし

戻り値	0～1	DMA ビジーステータス （1：DMA 動作中、0：DMA 停止中）

解説 本関数は、DMA 転送状態を返す。

注意 本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetDMACnt

RC 設定関数

DMA 転送カウントモニタ処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
void R_RC_GetDMACnt( ulong_t *pul_cnt );
```

引数			
	ulong_t *pul_cnt	O	Memory トランザクション転送回数へのポインタ （H’00_0001 : 1 回、H’FF_FFFF : 16,777,215 回、 H’00_0000 : 16,777,216 回）

戻り値 なし

解説 本関数は、DMA 転送開始からの Memory トランザクション転送数を返す。
転送数が 3 の場合であれば、2 回のトランザクション転送が完了し、3 回目のトランザクション転送中であることを示す。

注意 本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetRCIntEn

RC 設定関数

RC-DMA 割込み許可設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetRCIntEn( ushort_t us_int, uchar_t uc_mac_int, uchar_t uc_dl_int );
```

引数			
	ushort_t us_int	I	RC-DMA 割込みイネーブル（1：許可、0：無効） Bit8：Phyrdy 割込み Bit7：Link DLLACT 割込み Bit6：Link MAC 割込み Bit5：Link DL 割込み Bit4：受信コンプリーションエラー割込み Bit3：エラーパケット割込み Bit2：コンプリーションタイムアウト割込み Bit1：Unexpected TLP 受信割込み Bit0：DMA 転送正常終了割込み
	uchar_t uc_mac_int	I	MAC 割込みイネーブル（1：許可、0：無効） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
	uchar_t uc_dl_int	I	DL 割込みイネーブル（1：許可、0：無効） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、RC-DMA 割込みの許可/禁止を設定する。
Link 割込み（Bit6：MAC 割込み）の発生条件は MAC 割込みで指定する。
Link 割込み（Bit5：DL 割込み）の発生条件は DL 割込みで指定する。

注意

Unexpected TLP 受信割込みと DMA 転送正常終了割込みは、DMA 転送起動時のみ有効。
MAC 割込みイネーブル及び、DL 割込みイネーブルは、PCIe-Link 割込み関数と共通のため、双方からのアクセス時は注意のこと。

PCIe Phy 停止中の本関数コール時は、MAC 割込みイネーブル（uc_mac_int）及び、DL 割込みイネーブル（uc_dl_int）の設定は無効。

R_RC_GetRCIntEn

RC 設定関数

RC-DMA 割込み許可設定取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_GetRCIntEn( ushort_t *pus_int, uchar_t *puc_mac_int, uchar_t *puc_dl_int );
```

引数

ushort_t *pus_int	0	RC-DMA 割込みイネーブルへのポインタ（1：許可、0：無効） Bit8：Phyrdy 割込み Bit7：Link DLLACT 割込み Bit6：Link MAC 割込み Bit5：Link DL 割込み Bit4：受信コンプリーションエラー割込み Bit3：エラーパケット割込み Bit2：コンプリーションタイムアウト割込み Bit1：Unexpected TLP 受信割込み Bit0：DMA 転送正常終了割込み
uchar_t *puc_mac_int	0	MAC 割込みイネーブルへのポインタ（1：許可、0：無効） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
uchar_t *puc_dl_int	0	DL 割込みイネーブルへのポインタ（1：許可、0：無効） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、RC-DMA 割込みの許可/禁止設定を取得する。
Link 割込み（Bit6：MAC 割込み）の発生条件は MAC 割込みで指定する。
Link 割込み（Bit5：DL 割込み）の発生条件は DL 割込みで指定する。

注意

Unexpected TLP 受信割込みと DMA 転送正常終了割込みは、DMA 転送起動時のみ有効。
MAC 割込みイネーブル及び、DL 割込みイネーブルは、PCIe-Link 割込み関数と共通のため、双方からのアクセス時は注意のこと。

PCIe Phy 停止中の本関数コール時は、MAC 割込みイネーブル（*puc_mac_int）及び、DL 割込みイネーブル（*puc_dl_int）の取得は無効のため不定値で返す。

R_RC_ClrRCInt

RC 設定関数

RC-DMA 割込み要因クリア処理

書式

```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_ClrRCInt( ushort_t us_int );
```

引数

ushort_t us_int	I	RC-DMA 割込み要因（1 ライトクリア） Bit8：Phyrdy 割込み Bit7：Link DLLACT 割込み Bit6：Link MAC 割込み Bit5：Link DL 割込み Bit4：受信コンプリーションエラー割込み Bit3：エラーパケット割込み Bit2：コンプリーションタイムアウト割込み Bit1：Unexpected TLP 受信割込み Bit0：DMA 転送正常終了割込み	

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、RC-DMA 割込み要因のクリアを行う。

Link 割込み要因（Bit6：MAC 割込み）クリア時、以下の MAC 割込み要因も全てクリアする。
・Bit7：Receiver Error 割込み

Link 割込み要因（Bit5：DL 割込み）クリア時、以下の DL 割込み要因も全てクリアする。
・Bit5：DLLP Error 割込み
・Bit4：Replay TimeOut 割込み
・Bit3：Replay Number Roll Over 割込み
・Bit2：BAD TLP 割込み
・Bit1：BAD DLLP 割込み

注意

Unexpected TLP 受信割込みと DMA 転送正常終了割込みは、DMA 転送起動時のみ有効。
MAC 割込み要因及び、DL 割込み要因は、PCIe-Link 割込み関数と共通のため、双方からのアクセス時は注意のこと。

PCIe Phy 停止中の本関数コール時は、MAC 割込み要因及び、DL 割込み要因のクリアは無効。

R_RC_GetRCInt

RC 設定関数

RC-DMA 割込み要因取得処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_GetRCInt( ushort_t *pus_int, uchar_t *puc_mac_int, uchar_t *puc_dl_int );
```

引数			
	ushort_t *pus_int	0	RC-DMA 割込み要因へのポインタ（1：要因発生） Bit8：Phyrdy 割込み Bit7：Link DLLACT 割込み Bit6：Link MAC 割込み Bit5：Link DL 割込み Bit4：受信コンプリーションエラー割込み Bit3：エラーパケット割込み Bit2：コンプリーションタイムアウト割込み Bit1：Unexpected TLP 受信割込み Bit0：DMA 転送正常終了割込み
	uchar_t *puc_mac_int	0	MAC 割込み要因へのポインタ（1：要因発生） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
	uchar_t *puc_dl_int	0	DL 割込み要因へのポインタ（1：要因発生） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説　本関数は、RC-DMA 割込み要因を取得する。

注意　Unexpected TLP 受信割込みと DMA 転送正常終了割込みは、DMA 転送起動時のみ有効。
MAC 割込み要因及び、DL 割込み要因は、PCIe-Link 割込み関数と共通のため、双方からのアクセス時は注意のこと。

PCIe Phy 停止中の本関数コール時は、MAC 割込み要因（*puc_mac_int）及び、DL 割込み要因（*puc_dl_int）の取得は無効のため不定値で返す。

R_RC_SetLinkIntEn

RC 設定関数

Link 割込み許可設定処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_SetLinkIntEn( ushort_t us_int, ushort_t us_pm_int, uchar_t uc_mac_int, uchar_t uc_dl_int );
```

引数

ushort_t us_int	I	Link 割込みイネーブル（1：許可、0：無効） Bit12：Cpl タイムアウト割込み Bit11：Link ダウン割込み Bit9：エラーパケット受信割込み Bit8：Posted 受信割込み Bit7：NonPosted 受信割込み Bit6：Cpl 受信割込み Bit5：Posted 送信 Empty 割込み Bit4：NonPosted 送信 Empty 割込み Bit3：Cpl 送信 Empty 割込み Bit2：PM 割込み Bit1：MAC 割込み Bit0：DL 割込み
ushort_t us_pm_int	I	PM 割込みイネーブル（1：許可、0：無効） Bit15：L1 Fall Edge 割込み Bit7：PM Enter L1 割込み
uchar_t uc_mac_int	I	MAC 割込みイネーブル（1：許可、0：無効） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
uchar_t uc_dl_int	I	DL 割込みイネーブル（1：許可、0：無効） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Link 割込みの許可/禁止を設定する。
Link 割込み（Bit2：PM 割込み）の発生条件は PM 割込みで指定する。
Link 割込み（Bit1：MAC 割込み）の発生条件は MAC 割込みで指定する。
Link 割込み（Bit0：DL 割込み）の発生条件は DL 割込みで指定する。

注意

MAC 割込みイネーブル及び、DL 割込みイネーブルは、RC-DMA 割込み関数と共通のため、双方からのアクセス時は注意のこと。
本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetLinkIntEn

RC 設定関数

Link 割込み許可設定取得処理

書式

```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_GetLinkIntEn( ushort_t *pus_int, ushort_t *pus_pm_int, uchar_t *puc_mac_int, uchar_t *puc_dl_int );
```

引数

ushort_t *pus_int	O	Link 割込みイネーブルへのポインタ（1：許可、0：無効） Bit12：Cpl タイムアウト割込み Bit11：Link ダウン割込み Bit9：エラーパケット受信割込み Bit8：Posted 受信割込み Bit7：NonPosted 受信割込み Bit6：Cpl 受信割込み Bit5：Posted 送信 Empty 割込み Bit4：NonPosted 送信 Empty 割込み Bit3：Cpl 送信 Empty 割込み Bit2：PM 割込み Bit1：MAC 割込み Bit0：DL 割込み
ushort_t *pus_pm_int	O	PM 割込みイネーブルへのポインタ（1：許可、0：無効） Bit15：L1 Fall Edge 割込み Bit7：PM Enter L1 割込み
uchar_t *puc_mac_int	O	MAC 割込みイネーブルへのポインタ（1：許可、0：無効） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
uchar_t *puc_dl_int	O	DL 割込みイネーブルへのポインタ（1：許可、0：無効） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Link 割込みの許可/禁止設定を取得する。
Link 割込み（Bit2：PM 割込み）の発生条件は PM 割込みで指定する。
Link 割込み（Bit1：MAC 割込み）の発生条件は MAC 割込みで指定する。
Link 割込み（Bit0：DL 割込み）の発生条件は DL 割込みで指定する

注意

MAC 割込みイネーブル及び、DL 割込みイネーブルは、RC-DMA 割込み関数と共通のため、双方からのアクセス時は注意のこと。
本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_ClrLinkInt

RC 設定関数

Link 割込み要因クリア処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
char_t R_RC_ClrLinkInt( ushort_t us_int );
```

引数			
	ushort_t us_int	I	Link 割込み要因（1 ライトクリア） Bit12：Cpl タイムアウト割込み Bit11：DL_Down 割込み Bit9：エラーパケット受信割込み Bit8：Posted 受信割込み Bit7：NonPosted 受信割込み Bit6：Cpl 受信割込み Bit5：Posted 送信 Empty 割込み Bit4：NonPosted 送信 Empty 割込み Bit3：Cpl 送信 Empty 割込み Bit2：PM 割込み Bit1：MAC 割込み Bit0：DL 割込み

戻り値		
	RET_NORMAL	正常終了
	RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Link 割込み要因のクリアを行う。

LINK 割込み要因（Bit2：PM 割込み）クリア時、以下の PM 割込み要因も全てクリアする。
・Bit15：L1 Fall Edge 割込み
・Bit7：PM Enter L1 割込み

Link 割込み要因（Bit1：MAC 割込み）クリア時、以下の MAC 割込み要因も全てクリアする。
・Bit7：Receiver Error 割込み

Link 割込み要因（Bit0：DL 割込み）クリア時、以下の DL 割込み要因も全てクリアする。
・Bit5：DLLP Error 割込み
・Bit4：Replay TimeOut 割込み
・Bit3：Replay Number Roll Over 割込み
・Bit2：BAD TLP 割込み
・Bit1：BAD DLLP 割込み

注意

MAC 割込み要因及び、DL 割込み要因は、RC-DMA 割込み関数と共通のため、双方からのアクセス時は注意のこと。
本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_GetLinkInt

RC 設定関数

Link 割込み要因取得処理

書式

```
#include “r_common.h”
#include “iodefine.h”
#include “r_rc_if.h”
#include “rc_driver.h “
char_t R_RC_GetLinkInt( ushort_t *pus_int, ushort_t *pus_pm_int, uchar_t *puc_mac_int, uchar_t *puc_dl_int );
```

引数

ushort_t *pus_int	O	Link 割込み要因へのポインタ（1：要因発生） Bit12：Cpl タイムアウト割込み Bit11：Link ダウン割込み Bit9：エラーパケット受信割込み Bit8：Posted 受信割込み Bit7：NonPosted 受信割込み Bit6：Cpl 受信割込み Bit5：Posted 送信 Empty 割込み Bit4：NonPosted 送信 Empty 割込み Bit3：Cpl 送信 Empty 割込み Bit2：PM 割込み Bit1：MAC 割込み Bit0：DL 割込み
ushort_t *pus_pm_int	O	PM 割込み要因へのポインタ（1：要因発生） Bit15：L1 Fall Edge 割込み Bit7：PM Enter L1 割込
uchar_t *puc_mac_int	O	MAC 割込み要因へのポインタ（1：要因発生） Bit7：Receiver Error 割込み Bit6：Link Training 割込み
uchar_t *puc_dl_int	O	DL 割込み要因へのポインタ（1：要因発生） Bit7：DL Active 割込み Bit6：DL Down 割込み Bit5：DLLP Error 割込み Bit4：Replay TimeOut 割込み Bit3：Replay Number Roll Over 割込み Bit2：BAD TLP 割込み Bit1：BAD DLLP 割込み

戻り値

RET_NORMAL	正常終了
RET_ERR_LINK_ACCESS	LINK アクセスエラー

解説

本関数は、Link 割込み要因を取得する。

注意

MAC 割込み要因及び、DL 割込み要因は、RC-DMA 割込み関数と共通のため、双方からのアクセス時は注意のこと。

本関数は、PCIe Phy 停止中にコールしないこと。

R_RC_SetCallback

RC 設定関数

コールバック関数登録処理

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “

void R_RC_SetCallback( void ( *pv_rc_callback )( void ) );
```

引数

void (*pv_rc_callback)(void)	I	コールバック関数アドレス （NULL の場合、コールバックなし）

戻り値

なし

解説

本関数は、コールバック関数を登録する。

本関数で設定したコールバック関数は、PCIe 割込みハンドラ処理（R_RC_Interrupt）内でコールされる。

注意

R_RC_Interrupt

割り込みハンドラ関数

PCIe 割り込みハンドラ

書式

```
#include  “r_common.h”
#include  “iodefine.h”
#include  “r_rc_if.h”
#include  “rc_driver.h “
void R_RC_Interrupt( void );
```

引数　なし

戻り値　なし

解説　本関数は、PCIe 割り込み内での処理を行う。
PCIe 割込みハンドラから、本割込みハンドラ関数をコールすることにより、ユーザーが登録したコールバック関数をコールする。

注意　本関数は、PCIe 割り込みのハンドラ内からコールすること。

4. 動作条件

4.1 ポートについて

本 RC モジュールは PERST#（PCIe リセット信号）出力をサポートしていないため、RERST#出力が必要な場合は、SH7753 に搭載されている GPIO のポート制御で実現する。

4.2 割り込みについて

本デバイスドライバ使用時は、割り込みベクタテーブルに割り込みハンドラ関数を登録し、割り込み許可状態で使用すること。表 4に登録が必要な割り込みハンドラ関数を示す。

表4. 割り込みベクタテーブル設定関数一覧

割り込み要因番号	割り込み要因	割り込みハンドラ関数
0x27C0	RC	R_RC_Interrupt

4.3 使用メモリ領域について

本デバイスドライバで使用するメモリ領域を表 5に示す。また各ドライバ関数使用時に確保必要な構造体領域を表 6に示す。

表5. メモリ使用量一覧表

内容	セクション	属性	バイト数
プログラムコード	P	code, align=4	11,920 バイト
定数データ	C	data, align=4	88 バイト
初期値ありデータ	D	data, align=4	0 バイト
初期値なしデータ	B	data, align=4	4 バイト

表6. 使用構造体/共用体領域一覧表

構造体 typedef 名*	内容	バイト数	使用ドライバ関数
rc_link_stat_t	Link ステータス情報構造体	4 バイト	R_RC_GetLinkStatus
rc_rx_stat_t	受信ステータス情報構造体	4 バイト	R_RC_GetRxStatus
rc_tx_stat_t	送信ステータス情報構造体	1 バイト	R_RC_GetTxStatus
rc_err_packet_t	エラー要因情報共用体	1 バイト	R_RC_GetErrPacket
rc_tlp_h_t	ヘッダデータ共用体	16 バイト	R_RC_TxNP R_RC_TxP R_RC_TxCpl R_RC_RxNP R_RC_RxP R_RC_RxCpl R_RC_GetErrPacket
rc_dma_ctrl_t	DMA 転送管理構造体	32 バイト	R_RC_DMAStart

* RC ドライバのインタフェースヘッダ（r_rc_if.h）ファイルで定義している。

5. 注意事項

本 RC モジュールは、REFCLK（PCIe 基準クロック）の発振をサポートしていないため、外部（発振モジュール等）からの供給を必要とする。

6. 付録

6.1 ドライバ関数の使用例

RC ドライバのトランザクション送受信アクセス及び、初期設定の使用例を6.1.1～6.1.8に示す。

表7. ドライバ関数使用例一覧

6.1.1	リセット解除時の使用例
6.1.2	DL Active 時の使用例
6.1.3	NonPosted 送信時の使用例 ・Configration Read／Write 送信 ・IO Read／Write 送信 ・Memory Read 送信
6.1.4	Posted 送信時の使用例 ・Memory Write 送信 ・Message 送信
6.1.5	NonPosted 受信時の使用例 ・IO Read／Write 受信 ・Memory Read 受信
6.1.6	Posted 受信時の使用例 ・Memory Write 受信 ・Message 受信
6.1.7	DMA 転送時の使用例 ・Memory Read／Write 送信
6.1.8	PM 遷移/復帰時の使用例

6.1.1 リセット解除時の使用例

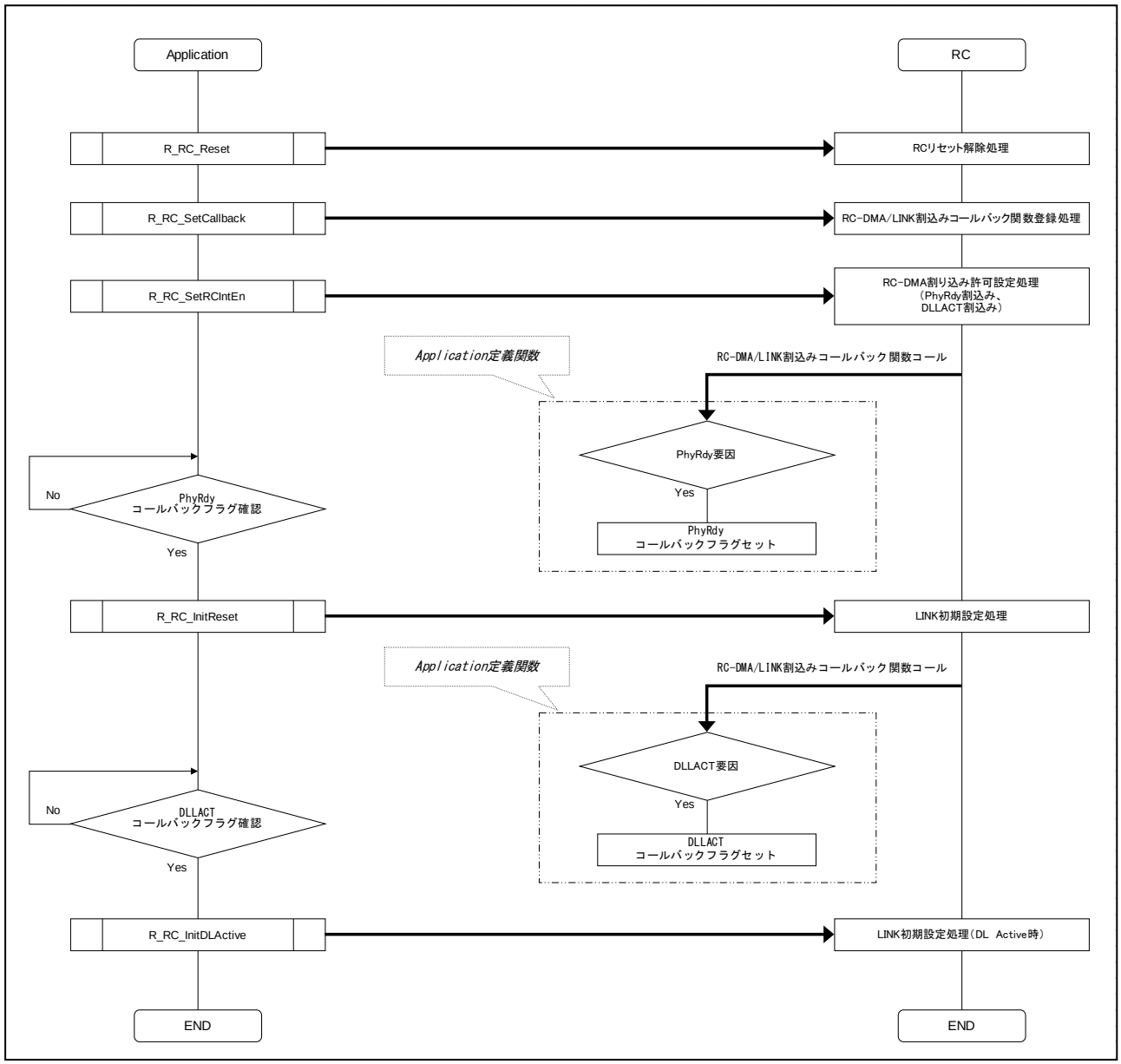

図 1. RC リセット解除時の使用例

6.1.2 DL Active 時の使用例

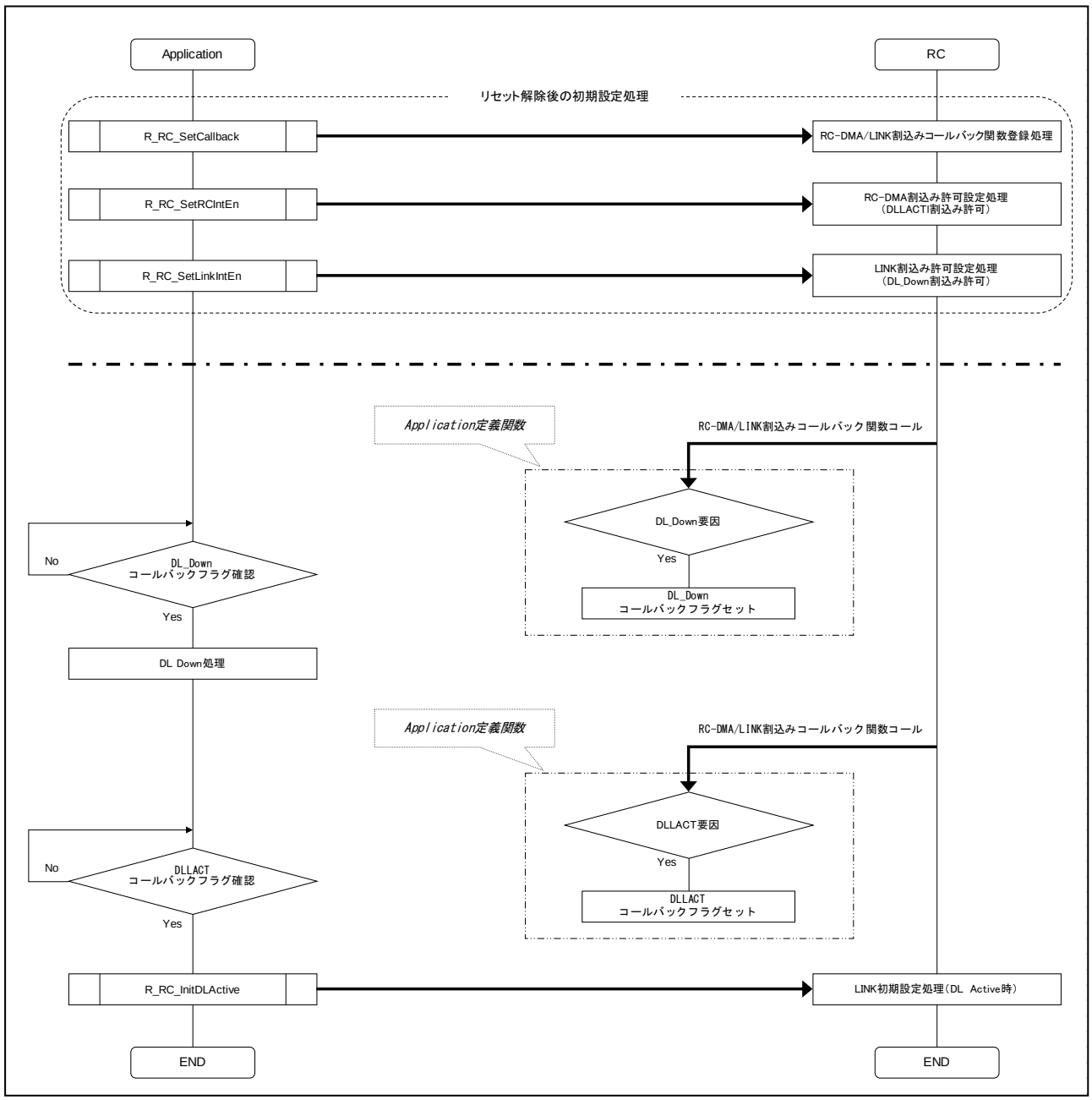

図 2. DL Active 時の使用例

6.1.3 NonPosted 送信時の使用例

Configration Read／Write 送信、IO Read／Write 送信、Memory Read 送信に使用する。

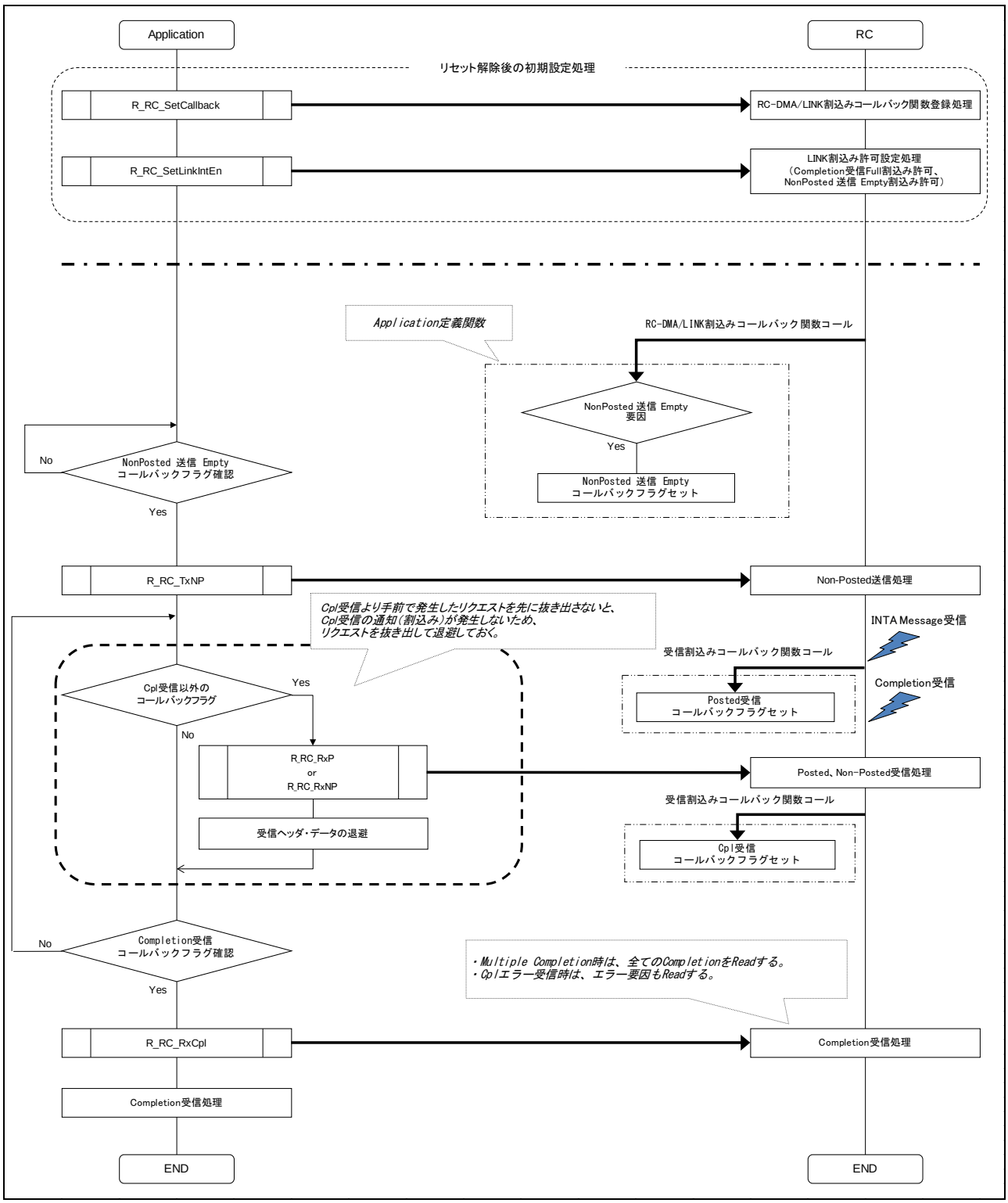

図 3. NonPosted 送信時の使用例

6.1.4 Posted 送信時の使用例

Memory Write 送信、Message 送信に使用する。

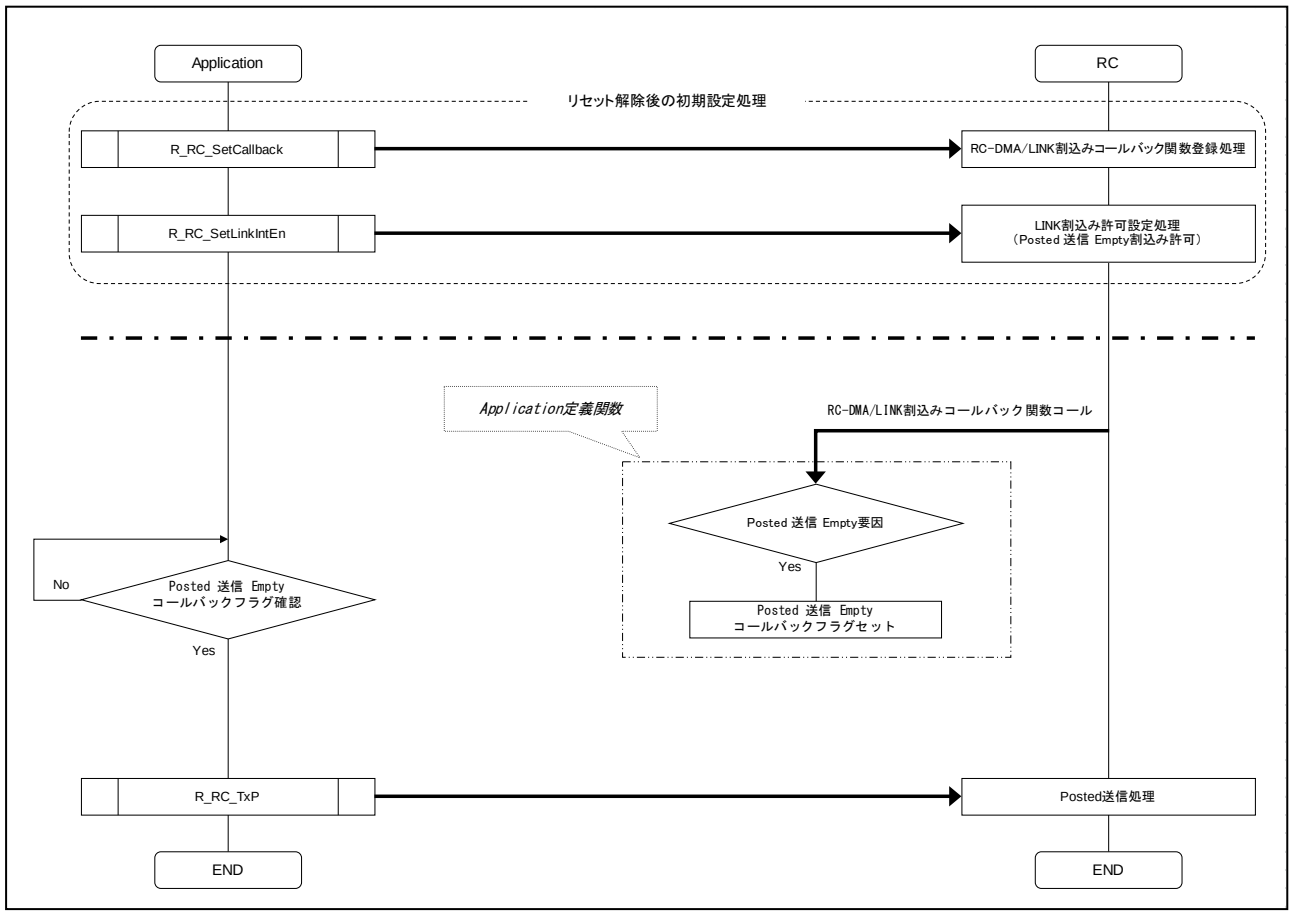

図 4. Posted 送信時の使用例

6.1.5 NonPosted 受信時の使用例

IO Read／Write 受信、Memory Read 受信に使用する。

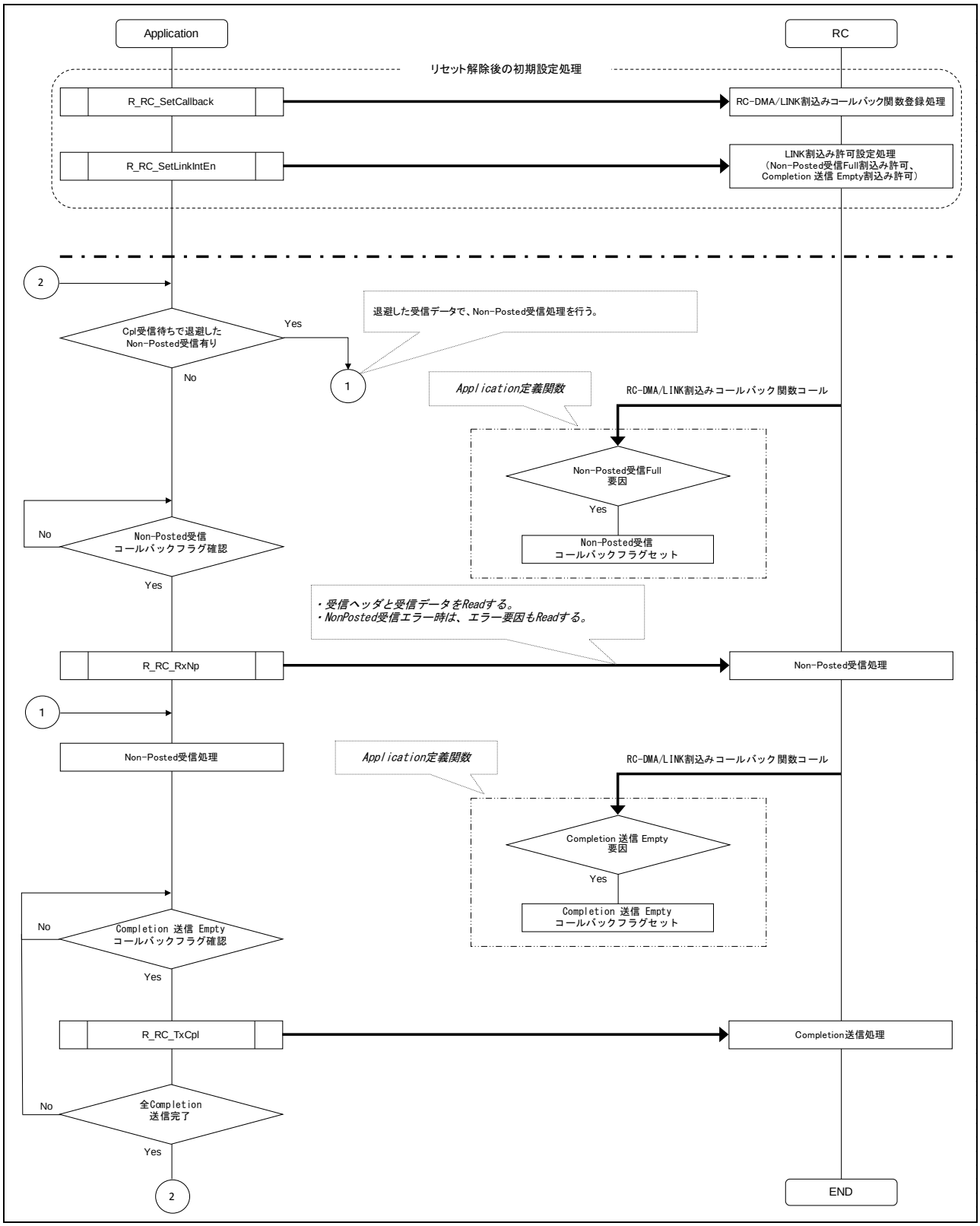

図 5. NonPosted 受信時の使用例

6.1.6 Posted 受信時の使用例

Memory Write 受信、Message 受信に使用する。

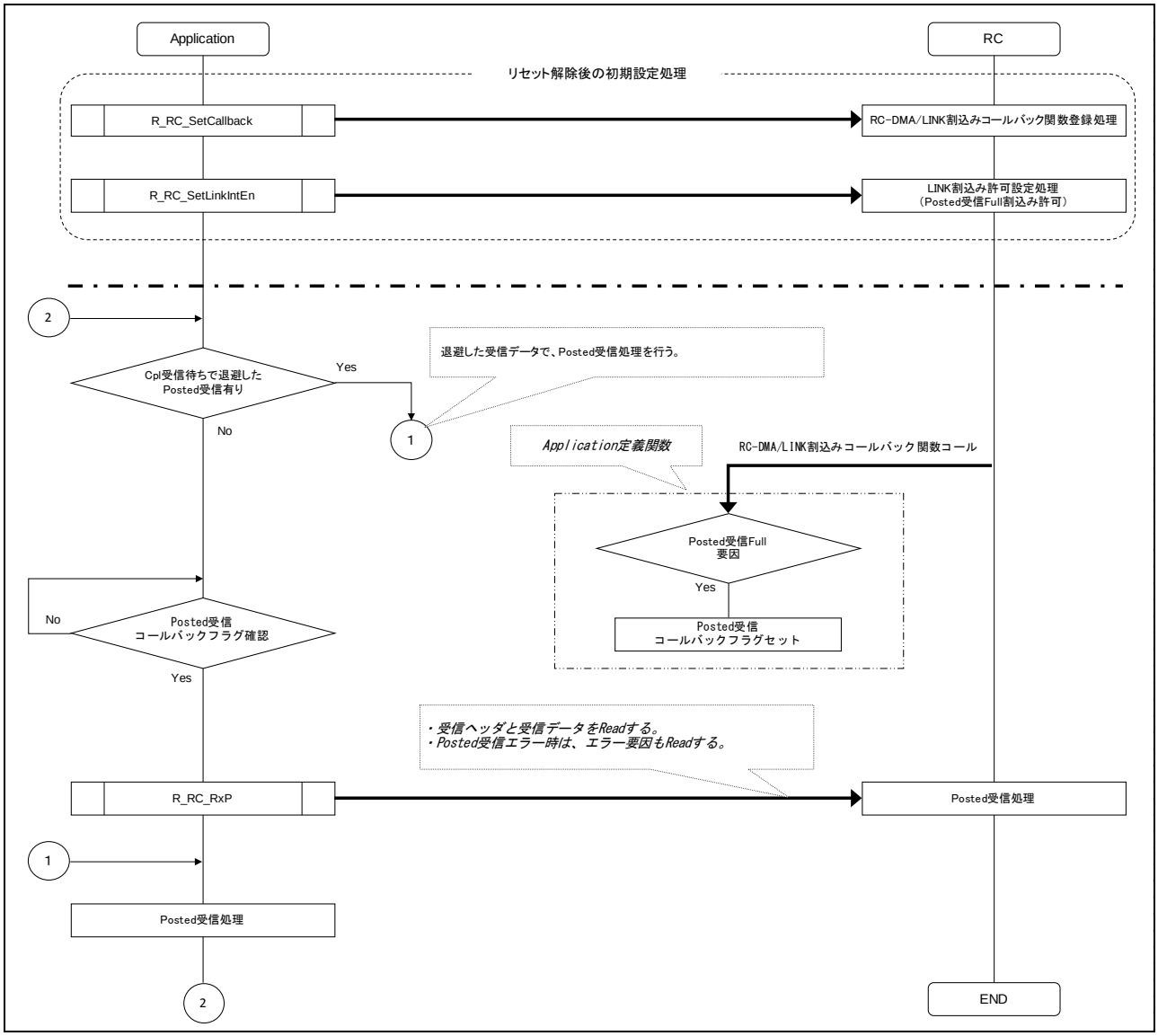

図 6. Posted 受信時の使用例

6.1.7 DMA 転送時の使用例

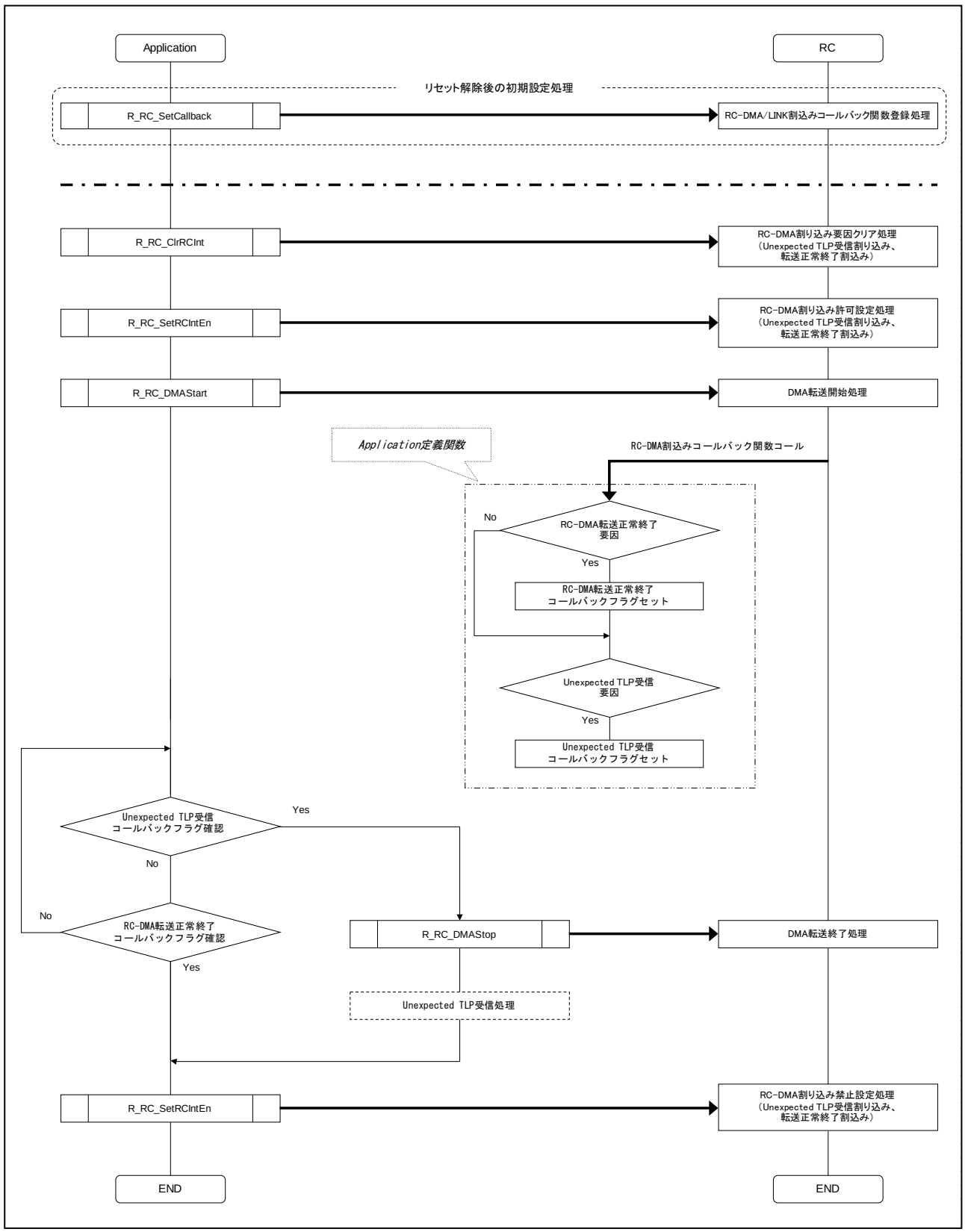

図 7. DMA 転送時の使用例

6.1.8 PM 遷移/復帰時の使用例

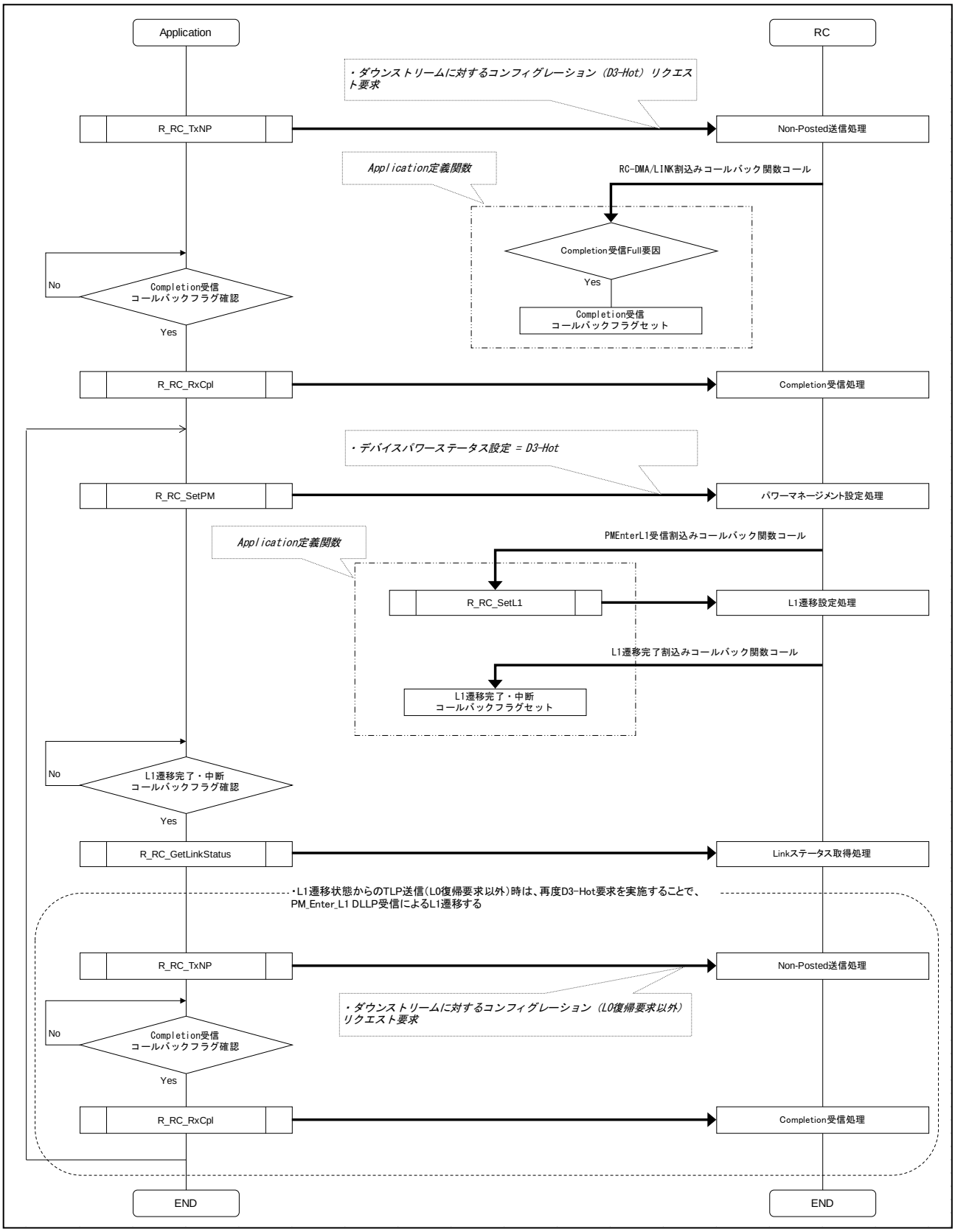

図 8. L1 遷移時の使用例

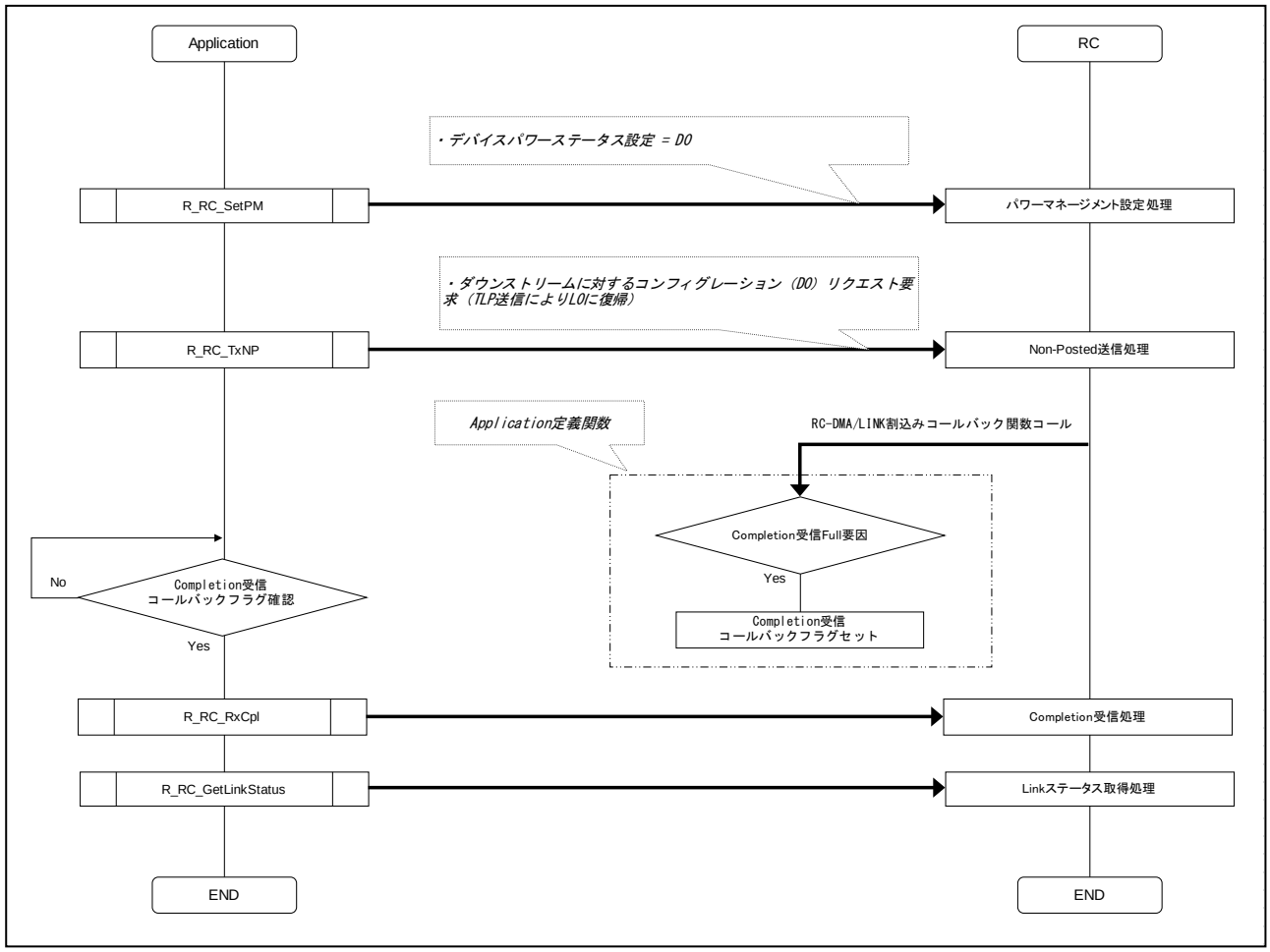

図 9. L0 復帰時の使用例

6.2 DMA 転送ヘッダ構成

R_RC_DMAStart 関数で指定するヘッダフォーマットは以下の通り。グレーの塗りつぶしは引数で指定した DMA 転送管理構造体（rc_dma_ctrl_t）に従い設定するため、ユーザが直接設定できないフィールド。

ビット	ビット名	値	R_RC_DMAStart 関数で行う処理
127-125	Format[2:0]	000b	Memory Read - 32bit 指定時 (pst_dma->ul_rcmemcfg.b.bit0=0、 pst_dma->ul_rcmemcfg.b.bit1=0)
		001b	Memory Read - 64bit 指定時 (pst_dma->ul_rcmemcfg.b.bit0=1、 pst_dma->ul_rcmemcfg.b.bit1=0)
		010b	Memory Write - 32bit 指定時 (pst_dma->ul_rcmemcfg.b.bit0=0、 pst_dma->ul_rcmemcfg.b.bit1=1)
		011b	Memory Write - 64bit 指定時 (pst_dma->ul_rcmemcfg.b.bit0=1、 pst_dma->ul_rcmemcfg.b.bit1=1)
124-120	Type	00000b	Memory Read/Write ともに 00000b で固定。
119	reserved	0b	固定値
118-116	Traffic Class	ユーザ設定値	
115	reserved	0b	固定値
114	Attr[2]	ユーザ設定値	ID-Based Ordering の指定
113	reserved	0b	固定値
112	TH	0b	TPH TLP はサポートしない。
111	TD (TLP Digest)	ユーザ設定値	ECRC 付加有無の指定
110	EP	ユーザ設定値	Data Poisoning の宣言
109-108	Attr[1:0]	ユーザ設定値	[1]は Relaxed Ordering の指定 [0]は No Snoop の指定
107-106	AT[1:0]	ユーザ設定値	Address Type の指定 00 : Default/Untranslated 01 : Translation Request 10 : Translated 11 : reserved
105-96	Length	001h	1DW (pst_dma->ul_rcmemcfg.b.bit6_4=000)
		002h	2DW (pst_dma->ul_rcmemcfg.b.bit6_4=001)
		004h	4DW (pst_dma->ul_rcmemcfg.b.bit6_4=010)
		008h	8DW (pst_dma->ul_rcmemcfg.b.bit6_4=011)

		010h	16DW (pst_dma->ul_rcmemcfg.b.bit6_4=100)
		020h	32DW (pst_dma->ul_rcmemcfg.b.bit6_4=101)
95-80	Requester ID	ユーザ設定値	Bus, Device, Function 番号
79-72	Tag	上位 3bit=000b, ユーザ設定値 5bit	Extended TAG 無効時 (pst_dma->ul_rcmemcfg.b.bit2=0) 5bit のユーザ設定値を TLP 送信のたびに自動インクリメント
		ユーザ設定値 8bit	Extended TAG 有効時 (pst_dma->ul_rcmemcfg.b.bit2=1) 8bit のユーザ設定値を TLP 送信のたびに自動インクリメント
71-68	Last BE	0000b	1DW 転送時 (pst_dma->ul_rcmemcfg.b.bit6_4=000)
		1111b	2DW/TLP 以上転送時 (pst_dma->ul_rcmemcfg.b.bit6_4=000 以外)
67-64	1st BE	1111b	Byte 転送はサポートしない。
32bit アドレッシング指定時 (pst_dma->ul_rcmemcfg.b.bit0=0) 開始アドレスを指定。TLP 送信のたびに自動インクリメント			
63-39	Address[31:7]	ユーザ設定値	転送アドレス
38	Address[6]	ユーザ設定値 / 0b	転送アドレス、32DW/TLP 以上転送時は 0 固定
37	Address[5]	ユーザ設定値 / 0b	転送アドレス、16DW/TLP 以上転送時は 0 固定
36	Address[4]	ユーザ設定値 / 0b	転送アドレス、8DW/TLP 以上転送時は 0 固定
35	Address[3]	ユーザ設定値 / 0b	転送アドレス、4DW/TLP 以上転送時は 0 固定
34	Address[2]	ユーザ設定値 / 0b	転送アドレス、2DW/TLP 以上転送時は 0 固定
33-32	Address[1:0]	00b	0 固定
31-0	reserved	0000_0000h	固定値
64bit アドレッシング指定時 (pst_dma->ul_rcmemcfg.b.bit0=1) 開始アドレスを指定。TLP 送信のたびに自動インクリメント			
63-7	Address[63:7]	ユーザ設定値	転送アドレス
6	Address[6]	ユーザ設定値 / 0b	転送アドレス、32DW/TLP 以上転送時は 0 固定
5	Address[5]	ユーザ設定値 / 0b	転送アドレス、16DW/TLP 以上転送時は 0 固定
4	Address[4]	ユーザ設定値 / 0b	転送アドレス、8DW/TLP 以上転送時は 0 固定
3	Address[3]	ユーザ設定値 / 0b	転送アドレス、4DW/TLP 以上転送時は 0 固定
2	Address[2]	ユーザ設定値 / 0b	転送アドレス、2DW/TLP 以上転送時は 0 固定
1-0	Address[1:0]	00b	0 固定

ホームページとサポート窓口

- ルネサス エレクトロニクスホームページ
  http://japan.renesas.com/
- お問合せ先
  http://japan.renesas.com/inquiry

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2013.7.1	ー	初版発行
1.01	2013.10.4	4	ドライバ改定によるスタックサイズ更新
		10	PCIe Phy 停止状態時の使用について注意事項を追加
		34-37	RC-DMA 割込み制御関数に DL, MAC 割込みを追加 PCIe Phy 停止状態時の使用について注意事項を追加
		38-41	Link 割込み制御関数に PM 割込みを追加 全送信バッファエンプティ割込みを削除
1.02	2014.2.14	24	メモリリードアドレス境界の制限事項を削除
		49	Cpl 応答待ちで Cpl 以外の TLP 受信が発生した場合、TLP を抜き取り、退避バッファに格納するフローを追加。
		51-52	退避バッファに TLP が存在する場合、退避した TLP の処理を実施するフローを追加。
1.03	2014.3.7	ー	サンプルプログラムコードの修正

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部 ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

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、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。
2. 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。
3. 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に関し、当社は、何らの責任を負うものではありません。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。
4. 当社製品を改造、改変、複製等しないでください。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）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   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（原子力制御システム、軍事機器等）に使用されることを意図しておらず、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お客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の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。
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。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。
9. 本資料に記載されている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に使用しないでください。当社製品または技術を輸出する場合は、「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。
10. お客様の転売等により、本ご注意書き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、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
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。

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 エレクトロニクス株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル）

■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/contact/