R20TS0025JJ0100
Rev.1.00
2016.05.16 号

【注意事項】
RX ファミリ
パラレルデータキャプチャユニット（PDC）モジュール
Firmware Integration Technology
RX64M グループ RX Driver Package Ver.1.01
RX64M, RX71M グループ RX Driver Package Ver.1.02

概要

RX ファミリ パラレルデータキャプチャユニット（PDC）モジュール Firmware Integration Technology（FIT）、RX64M グループ RX Driver Package Ver.1.01 および RX64M, RX71M グループ RX.1.02 の使用上の注意事項を連絡します。

1. PDC FIT モジュールの“callback_frame_end()” および “error_processing()” 関数におけるフラグ処理の注意事項

1. PDC FIT モジュールの“callback_frame_end()”および“error_processing()” 関数におけるフラグ処理の注意事項

1.1 該当製品

- RX ファミリ パラレルデータキャプチャユニット（PDC）モジュール
  Firmware Integration Technology（以下、PDC FIT モジュール）

  対応リビジョン: Rev1.00, Rev1.01, Rev1.02

  対応するドキュメントは以下です。

  ・RX ファミリ パラレルデータキャプチャユニット（PDC）モジュール
  Firmware Integration Technology アプリケーションノート

  資料番号: R01AN2220JJ0100（Rev1.00 対応）

  資料番号: R01AN2220JJ0101（Rev1.01 対応）

  資料番号: R01AN2220JJ0102（Rev1.02 対応）

上記 PDC FIT モジュールが同梱[注1、注2]されている下記製品も対象になります。

- RX64M グループ RX Driver Package Ver.1.01

  注 1: r_pdc_rx_v1.00.zip として PDC FIT モジュールが同梱されています。

  対応するドキュメントは以下です。

  ・RX64M グループ RX Driver Package ユーザーズマニュアル アプリケーションノート

  資料番号: R01AN2460JJ0101

- RX64M, RX71M グループ RX Driver Package Ver.1.02

  注 2: r_pdc_rx_v1.02.zip として PDC FIT モジュールが同梱されています。

  対応するドキュメントは以下です。

  ・RX64M, RX71M グループ RX Driver Package Ver.1.02 アプリケーションノート

  資料番号: R01AN2606JJ0102

  資料番号: R01AN2606JJ0103

1.2 該当 MCU

RX64M グループおよび RX71M グループ

1.3 発生条件

以下のいずれかの条件で使用している場合に発生します。

(1) PDC FIT モジュールの API 関数 “R_PDC_Open()” を実行して “callback_frame_end()” および “callback_error()” を割り込み発生時のコールバック関数として登録している。

(2) PDC FIT モジュールのアプリケーションノート（R01AN2220JJ0100、R01AN2220JJ0101 および R01AN2220JJ0102）の「4.1 API 使用例」に従い “pdc_pcfei_callback()” および “pdc_error_processing()” を割り込み発生時のコールバック関数として登録している。

1.4 内容

(1) 1.3 発生条件(1)の場合

PDC FIT モジュールのソースコード “r_pdc_rx.c” 内の “callback_frame_end()” および “error_processing()” 関数のフラグ処理の内容に誤りがあるため、以下の不具合が発生しアプリケーションプログラムの処理が正常に実行されません。

(2) 1.3 発生条件(2)の場合

PDC FIT モジュールのアプリケーションノート（R01AN2220JJ0100、R01AN2220JJ0101 および R01AN2220JJ0102）「4.1 API 使用例 (2) サンプルコード」の“pdc_pcfei_callback()” および “pdc_error_processing()” 関数のフラグ処理の内容に誤りがあるため、以下の不具合が発生しアプリケーションプログラムの処理が正常に実行されません。

- 不具合現象

・フレームエンド割り込み発生時にオーバランエラー、アンダランエラー、垂直方向ライン数設定エラーおよび水平方向バイト数設定エラーのいずれかのエラーが発生している場合、フレームエンドフラグをクリアする時に他のエラーフラグもクリアされます。そのため以降のエラー処理が実行されません。

・エラー割り込み発生時にオーバランエラー、アンダランエラー、垂直方向ライン数設定エラーおよび水平方向バイト数設定エラーで複数のエラーが発生している場合、最初のエラーフラグをクリアする時に他のエラーフラグもクリアされます。

1.5 回避策

(1) 1.3 発生条件(1)の場合

以下の関数のフラグ処理を正しい処理に変更してください。

・PDC FIT モジュールのソースコード “r_pdc_rx.c” の “callback_frame_end()” および “error_processing()” 関数

以下に、変更内容の詳細を記します。各関数の青文字の処理を赤文字の処理に変更してください。

備考: 回避策では R_PDC_Control 関数をステータスクリアコマンドで実行する処理で指定する引数を新たに宣言しています。また、R_PDC_Control 関数を呼び出す前に引数にクリアするフラグを”true”に、クリアしないフラグを”false”に設定しています。

■ PDC FIT モジュールのソースコード “r_pdc_rx.c” の“callback_frame_end()” 関数

修正前: フラグ処理（2 箇所）抜粋

```
static void callback_frame_end(void)
{
    volatile pdc_return_t ret_pdc;
    pdc_data_cfg_t dummy_data;
    pdc_stat_t stat_pdc;
(中略)
        /* Clear the PCSR.FEF flag to 0. */
        stat_pdc.pcsr_stat.fifo_empty = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
    /* Clear the FEF flag in the PCSR register to 0. */
    stat_pdc.pcsr_stat.fifo_empty = true;
    ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
    if (PDC_SUCCESS != ret_pdc)
```

修正後:

```
static void callback_frame_end(void)
{
    volatile pdc_return_t ret_pdc;
    pdc_data_cfg_t dummy_data;
    pdc_stat_t stat_pdc;
    pdc_stat_t clear_stat_pdc;
(中略)
        /* Clear the PCSR.FEF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = true;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
    /* Clear the FEF flag in the PCSR register to 0. */
    clear_stat_pdc.pcsr_stat.frame_busy = false;
    clear_stat_pdc.pcsr_stat.fifo_empty = false;
    clear_stat_pdc.pcsr_stat.frame_end = true;
    clear_stat_pdc.pcsr_stat.overrun = false;
    clear_stat_pdc.pcsr_stat.underrun = false;
    clear_stat_pdc.pcsr_stat.verf_error = false;
    clear_stat_pdc.pcsr_stat.herf_error = false;
    ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &clear_stat_pdc);
    if (PDC_SUCCESS != ret_pdc)
```

■ PDC FIT モジュールのソースコード“r_pdc_rx.c” の“error_processing()” 関数

修正前: フラグ処理（4 箇所）抜粋

```
static void error_processing(void)
{
    volatile pdc_return_t ret_pdc;
    pdc_data_cfg_t dummy_data;
    pdc_stat_t stat_pdc;
(中略)
        /* Clear the PCSR.OVRF flag to 0. */
        stat_pdc.pcsr_stat.overrun = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.UDRF flag to 0. */
        stat_pdc.pcsr_stat.underrun = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.VERF flag to 0. */
        stat_pdc.pcsr_stat.verf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.HERF flag to 0. */
        stat_pdc.pcsr_stat.herf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
         if (PDC_SUCCESS != ret_pdc)
```

修正後:

```
static void error_processing(void)
{
    volatile pdc_return_t ret_pdc;
    pdc_data_cfg_t dummy_data;
    pdc_stat_t stat_pdc;
    pdc_stat_t clear_stat_pdc;
(中略)
        /* Clear the PCSR.OVRF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = true;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.UDRF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = true;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.VERF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = true;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.HERF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
         if (PDC_SUCCESS != ret_pdc)
```

(2) 1.3 発生条件(2)の場合

以下の関数のフラグ処理を正しい処理に変更してください。

・PDC FIT モジュールのアプリケーションノート（R01AN2220JJ0100、R01AN2220JJ0101 および R01AN2220JJ0102）「4.1 API 使用例 (2) サンプルコード」の“pdc_pcfei_callback()” および “pdc_error_processing()” 関数

以下に、変更内容の詳細を記します。各関数の青文字の処理を赤文字の処理に変更してください。

備考: 回避策では R_PDC_Control 関数をステータスクリアコマンドで実行する処理で指定する引数を新たに宣言しています。また、R_PDC_Control 関数を呼び出す前に引数にクリアするフラグを”true”に、クリアしないフラグを”false”に設定しています。

■ PDC FIT モジュールのアプリケーションノート（R01AN2220JJ0100、R01AN2220JJ0101 および R01AN2220JJ0102）「4.1 API 使用例 (2) サンプルコード」“pdc_pcfei_callback()” 関数

修正前: フラグ処理（2 箇所）抜粋

```
void pdc_pcfei_callback(void)
{
    volatile pdc_return_t ret_pdc; /* PDC API のエラーコード */
    pdc_data_cfg_t dummy_data;     /* 使用しない */
    pdc_stat_t stat_pdc;           /* PDC ステータス */
(中略)
        /* Clear the PCSR.FEF flag to 0. */
        stat_pdc.pcsr_stat.fifo_empty = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the FEF flag in the PCSR register to 0. */
        stat_pdc.pcsr_stat.fifo_empty = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
```

修正後:

```
void pdc_pcfei_callback(void)
{
    volatile pdc_return_t ret_pdc; /* PDC API のエラーコード */
    pdc_data_cfg_t dummy_data;     /* 使用しない */
    pdc_stat_t stat_pdc;           /* PDC ステータス */
    pdc_stat_t clear_stat_pdc;
(中略)
        /* Clear the PCSR.FEF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = true;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
    /* Clear the FEF flag in the PCSR register to 0. */
    clear_stat_pdc.pcsr_stat.frame_busy = false;
    clear_stat_pdc.pcsr_stat.fifo_empty = false;
    clear_stat_pdc.pcsr_stat.frame_end = true;
    clear_stat_pdc.pcsr_stat.overrun = false;
    clear_stat_pdc.pcsr_stat.underrun = false;
    clear_stat_pdc.pcsr_stat.verf_error = false;
    clear_stat_pdc.pcsr_stat.herf_error = false;
    ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &clear_stat_pdc);
    if (PDC_SUCCESS != ret_pdc)
```

■ PDC FIT モジュールのアプリケーションノート（R01AN2220JJ0100、R01AN2220JJ0101 および R01AN2220JJ0102）「4.1 API 使用例 (2) サンプルコード」“pdc_error_processing()” 関数

修正前: フラグ処理（4 箇所）抜粋

```
void pdc_error_processing(void)
{
    volatile pdc_return_t ret_pdc; /* PDC API のエラーコード */
    pdc_data_cfg_t dummy_data;        /* 使用しない */
    pdc_stat_t stat_pdc;                /* PDC ステータス */
(中略)
        /* Clear the PCSR.OVRF flag to 0. */
        stat_pdc.pcsr_stat.overrun = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.UDRF flag to 0. */
        stat_pdc.pcsr_stat.underrun = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.VERF flag to 0. */
        stat_pdc.pcsr_stat.verf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.HERF flag to 0. */
        stat_pdc.pcsr_stat.herf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data, &stat_pdc);
         if (PDC_SUCCESS != ret_pdc)
```

修正後:

```
void pdc_error_processing(void)
{
    volatile pdc_return_t ret_pdc; /* PDC API のエラーコード */
    pdc_data_cfg_t dummy_data;        /* 使用しない */
    pdc_stat_t stat_pdc;                /* PDC ステータス */
    pdc_stat_t clear_stat_pdc;
(中略)
        /* Clear the PCSR.OVRF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = true;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.UDRF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = true;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.VERF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = true;
        clear_stat_pdc.pcsr_stat.herf_error = false;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
        if (PDC_SUCCESS != ret_pdc)
(中略)
        /* Clear the PCSR.HERF flag to 0. */
        clear_stat_pdc.pcsr_stat.frame_busy = false;
        clear_stat_pdc.pcsr_stat.fifo_empty = false;
        clear_stat_pdc.pcsr_stat.frame_end = false;
        clear_stat_pdc.pcsr_stat.overrun = false;
        clear_stat_pdc.pcsr_stat.underrun = false;
        clear_stat_pdc.pcsr_stat.verf_error = false;
        clear_stat_pdc.pcsr_stat.herf_error = true;
        ret_pdc = R_PDC_Control(PDC_CMD_STATUS_CLR, &dummy_data,
        &clear_stat_pdc);
         if (PDC_SUCCESS != ret_pdc)
```

1.6 恒久対策

- RX ファミリ パラレルデータキャプチャユニット（PDC）モジュール
  Firmware Integration Technology

  次期バージョンで改修予定です。（2016 年 5 月予定）

- RX64M グループ RX Driver Package Ver.1.01 および RX64M, RX71M グループ RX Driver Package Ver.1.02

  次期バージョン（Ver.1.10）にて本注意事項を改修した PDC FIT モジュールを同梱予定です。
  （2016 年 7 月予定）

以上

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.05.16	-	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
http://www.renesas.com/ja-jp/support/contact.html