RZ/A2M グループ

Video Display Controller and Sprite Engine Sample Driver

R01AN4475JJ0101
Rev.1.01
2018.12.28

要旨

本書は RZ/A シリーズの RZ/A2M のビデオディスプレイコントローラ 6(VDC)及びスプライトエンジン(SPEA)のサンプルドライバについて説明します。

動作確認デバイス

RZ/A2M

目次

1. 仕様

本ドライバは、RZ/A2M グループ内蔵の VDC を利用し映像入力や表示の制御を行います。

本ドライバが対応している機能を以下に示します。

表 1-1　VDC ドライバの機能

項目	機能
入力映像規格	ITU-R BT.656 規格準拠 8bit (27MHz、インタレース信号)に対応 ITU-R BT.656 規格拡張 8bit (27MHz、プログレッシブ信号)に対応 ※ ITU-R BT.601 規格拡張 8bit (27MHz、インタレース信号)に対応 ※ ITU-R BT.601 規格拡張 8bit (54MHz、プログレッシブ信号)に対応 ※ ITU-R BT.601 規格拡張 16bit (13.5MHz、インタレース信号) ※ デジタル端子入力：YCbCr422、YCbCr444、RGB888、RGB666、RGB565 映像に対応
映像録画	YCbCr422/YCbCr444/RGB565/RGB888 形式 1/1、1/2、1/4、1/8 フィールドのレートで映像を保存
映像画質調整	コントラスト調整、ブライト調整、水平ノイズリダクション、黒伸張、LTI／シャープネス
映像スケーリング／回転	垂直／水平スケーリング: 1/8 ~ 8 倍 0、90、180、270 度回転および水平鏡像
グラフィックスプレーン	グラフィックスプレーン: 3 面 対応フォーマット: RGB565、RGB888、ARGB1555、ARGB4444、ARGB8888、RGBA5551、RGBA8888、CLUT8、CLUT4、CLUT1、YCbCr422、YCbCr444
グラフィックス機能	矩形領域アルファブレンディング（フェード機能あり） クロマキー 画素単位アルファブレンディング
出力映像サイズ	出力映像サイズ例: XGA (1024x768)、SVGA (800x600)、WVGA (800x480)、VGA (640x480)、WQVGA (480x240)、QVGA 横長 (320x240)、QVGA 縦長 (240x320)
出力映像形式	プログレッシブ映像出力 • RGB888 (24bit のパラレル出力) • RGB666 (18bit のパラレル出力) • RGB565 (16bit のパラレル出力) • RGB888 (8bit のシリアル出力)
パネル出力調整	パネルブライト、コントラスト調整、RGB ガンマ補正、ディザ処理、出力フォーマット変換

【注】 ITU-R BT.656、601 にプログレッシブに関する記述はありません。また、ITU-R BT.601 に接続インタフェースに関する記述はありません。

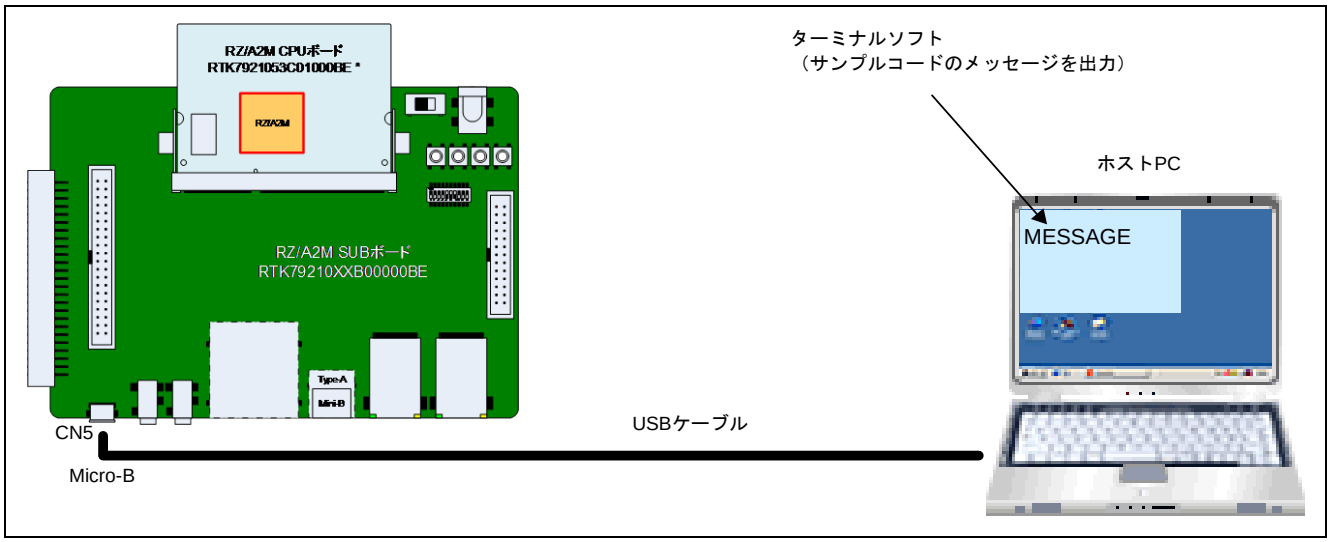

RZ/A2M CPUボード
RTK7921053C01000BE
RZ/A2M SUBボード
RTK79210XXB00000BE
CN5
Micro-B
USBケーブル
ターミナルソフト
（サンプルコードのメッセージを出力）
ホストPC
MESSAGE

図 1.1　動作環境

2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表 2.1　動作確認条件

項目	内容
使用マイコン	RZ/A2M
動作周波数（注）	CPU クロック（Iφ）：528MHz 画像処理クロック（Gφ）：264MHz 内部バスクロック（Bφ）：132MHz 周辺クロック 1（P1φ）：66MHz 周辺クロック 0（P0φ）：33MHz QSPI0_SPCLK：66MHz CKIO：132MHz
動作電圧	電源電圧（I/O）：3.3V 電源電圧（1.8/3.3V 切替 I/O（PVcc_SPI））：3.3V 電源電圧（内部）：1.2V
統合開発環境	e2 studio V7.3.0
C コンパイラ	GNU Arm Embedded Toolchain 6-2017-q2-update
	コンパイラオプション（ディレクトリパスの追加は除く） Release: -mcpu=cortex-a9 -march=armv7-a -marm -mthumb-interwork -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Os -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -Wstack-usage=100 -fabi-version=0 Hardware Debug: -mcpu=cortex-a9 -march=armv7-a -marm -mthumb-interwork -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Og -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -g3 -Wstack-usage=100 -fabi-version=0
動作モード	ブートモード 3 （シリアルフラッシュブート 3.3V 品）
ターミナルソフトの通信設定	• 通信速度：115200bps • データ長：8 ビット • パリティ：なし • ストップビット長：1 ビット • フロー制御：なし
使用ボード	RZ/A2M CPU ボード　RTK7921053C00000BE RZ/A2M SUB ボード　RTK79210XXB00000BE
使用デバイス （ボード上で使用する機能）	• シリアルフラッシュメモリ（SPI マルチ I/O バス空間に接続） メーカ名：Macronix 社、型名：MX25L51245GXD • RL78/G1C（USB 通信とシリアル通信を変換し、ホスト PC との通信に使用）

【注】 クロックモード 1（EXTAL 端子からの 24MHz のクロック入力）で使用時の動作周波数です。

3. 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

特になし

4. ハードウェア説明

4.1 ハードウェア構成例

ハードウェア構成例については RZ/A2M 評価ボードのマニュアルを参照ください。

4.2 使用端子一覧

使用端子と機能を、表 4-1 に示します。

表 4-1　使用端子と機能

端子名	入出力	内容	RZ/A2M 評価ボード接続
DV0_CLK	入力	外部入力クロック 0	未使用
DV0_VSYNC	入力	外部入力垂直同期 0	未使用
DV0_HSYNC	入力	外部入力水平同期 0	未使用
DV0_DATA23~0	入力	外部入力映像データ 0	未使用
LCD0_CLK	出力	パネルクロック 0	PJ_6
LCD0_DATA23~0	出力	パネル用映像データ 0	PB_5-0, PA_7-0, P8_0, PF_7-0, PH_2
LCD0_TCON6~0	出力	パネル用制御信号 0	PC_3(TCON4), PC_4(TCON3), P7_7(TCON0)
LCD0_EXTCLK	入力	パネルクロックソース 0	PJ_6
TXCLKOUTM/P	出力	LVDS クロック出力端子	P4_7, P4_6
TXOUT2M/P	出力	LVDS データ出力端子	P4_5, P4_4
TXOUT1M/P	出力	LVDS データ出力端子	P4_3, P4_2
TXOUT0M/P	出力	LVDS データ出力端子	P4_1, P4_0

5. ソフトウェア説明

5.1 ファイル構成

表 5-1、表 5-2 に本ドライバを構成するファイルを示します。

表 5-1　VDC ドライバのファイル構成

ファイル名	説明
r_vdc.c	VDC ドライバ API 関数 VDC ドライバの API 関数を記述したソースファイル。
r_vdc.h	VDC ドライバ API 定義 VDC ドライバ API 関数のプロトタイプや、API として定義されたパラメータについて記述したヘッダファイル。
r_vdc_check_parameter.c	VDC ドライバパラメータチェック処理 VDC ドライバのパラメータチェック処理を記述したソースファイル。
r_vdc_check_parameter.h	VDC ドライバパラメータチェック定義 VDC ドライバのパラメータチェック処理関数のプロトタイプを記述したヘッダファイル。
r_vdc_interrupt.c	VDC ドライバ割り込み関連処理 VDC の割り込み関連の設定処理と割り込みサービスルーチンを記述したソースファイル。
r_vdc_register.c	VDC ドライバレジスタ設定処理 VDC のレジスタ設定処理を記述したソースファイル。
r_vdc_register.h	VDC ドライバレジスタ設定定義 VDC のレジスタ設定処理関数のプロトタイプ、及びレジスタアドレステーブルの構造体を記述したヘッダファイル。
r_vdc_register_address.c	VDC ドライバレジスタアドレステーブル VDC のレジスタアドレスを格納したテーブルを記述したファイル。
r_vdc_shared_param.c	VDC ドライバ共有パラメータ処理 VDC ドライバ内部で共有するパラメータの設定、取得処理を記述したソースファイル。
r_vdc_shared_param.h	VDC ドライバ共有パラメータ定義 VDC ドライバの共有するパラメータ設定／取得処理関数のプロトタイプを記述したヘッダファイル。
r_vdc_user.h	VDC ドライバユーザ定義ヘッダ コンパイルスイッチや、ユーザが静的に修正可能な定数を定義するヘッダファイル。

表 5-2　SPEA ドライバのファイル構成

ファイル名	説明
r_spea.c	SPEA ドライバ API 関数 SPEA ドライバの API 関数を記述したソースファイル。
r_spea.h	SPEA ドライバ API 定義 SPEA ドライバ API 関数のプロトタイプや、API として定義されたパラメータについて記述したヘッダファイル。
r_spea_check_parameter.c	SPEA ドライバパラメータチェック処理 SPEA ドライバのパラメータチェック処理を記述したソースファイル。
r_spea_check_parameter.h	SPEA ドライバパラメータチェック定義

	SPEA ドライバのパラメータチェック処理関数のプロトタイプを記述したヘッダファイル。
r_spea_register.c	SPEA ドライバレジスタ設定処理 SPEA レジスタ設定処理を記述したソースファイル。
r_spea_register.h	SPEA ドライバレジスタ設定定義 SPEA のレジスタ設定処理関数のプロトタイプ、及びレジスタアドレステーブルの構造体を記述したヘッダファイル。
r_spea_register_address.c	SPEA ドライバレジスタアドレステーブル SPEA のレジスタアドレスを格納したテーブルを記述したファイル。
r_spea_user.h	SPEA ドライバユーザ定義ヘッダ コンパイルスイッチや、ユーザが静的に修正可能な定数を定義するヘッダファイル。

また、本ドライバでは以下の外部ファイルを参照しています。

表 5-3　VDC, SPEA ドライバの参照する外部ファイル

ファイル名	説明
r_typedefs.h	基本型定義ヘッダ 基本型を定義するヘッダファイル。
iodefine.h	I/O 定義ヘッダ I/O 定義を含むヘッダファイル。

5.2　列挙型定義

以下に列挙型定義について記載します。エラーコードについては「5.4 エラーコード」を参照ください。

(1)　vdc_channel_t

vdc_channel_t は VDC のチャネルを表す列挙型です。

```
typedef enum
{
    VDC_CHANNEL_0 = 0,
    VDC_CHANNEL_NUM
} vdc_channel_t;
```

列挙定数	値	説明
VDC_CHANNEL_0	0	チャネル 0
VDC_CHANNEL_NUM	1	チャネル数

(2)　vdc_onoff_t

vdc_onoff_t は ON と OFF を表す列挙型です。

```
typedef enum
{
    VDC_OFF    = 0,
    VDC_ON     = 1
} vdc_onoff_t;
```

列挙定数	値	説明
VDC_OFF	0	OFF
VDC_ON	1	ON

(3)　vdc_edge_t

vdc_edge_t は信号のエッジを表す列挙型です。

```
typedef enum
{
    VDC_EDGE_RISING    = 0,
    VDC_EDGE_FALLING   = 1
} vdc_edge_t;
```

列挙定数	値	説明
VDC_EDGE_RISING	0	立ち上がりエッジ
VDC_EDGE_FALLING	1	立ち下りエッジ

(4)　vdc_sig_pol_t

vdc_sig_pol_t は信号の極性を表す列挙型です。

```
typedef enum
{
    VDC_SIG_POL_NOT_INVERTED    = 0,
    VDC_SIG_POL_INVERTED      = 1
} vdc_sig_pol_t;
```

列挙定数	値	説明
VDC_SIG_POL_NOT_INVERTED	0	非反転
VDC_SIG_POL_INVERTED	1	反転

(5) vdc_scaling_type_t

vdc_scaling_type_t はスケーリング部のタイプを表す列挙型です。

```
typedef enum
{
    VDC_SC_TYPE_SC0 = 0,
    VDC_SC_TYPE_NUM
} vdc_scaling_type_t;
```

列挙定数	値	説明
VDC_SC_TYPE_SC0	0	スケーリング部 0
VDC_SC_TYPE_NUM	1	スケーリング部のタイプ数

(6) vdc_graphics_type_t

vdc_graphics_type_t はグラフィックス部のタイプを表す列挙型です。

```
typedef enum
{
    VDC_GR_TYPE_GR0 = 0,
    VDC_GR_TYPE_GR2,
    VDC_GR_TYPE_GR3,
    VDC_GR_TYPE_NUM
} vdc_graphics_type_t;
```

列挙定数	値	説明
VDC_GR_TYPE_GR0	0	グラフィックス部 0
VDC_GR_TYPE_GR2	1	グラフィックス部 2
VDC_GR_TYPE_GR3	2	グラフィックス部 3
VDC_GR_TYPE_NUM	3	グラフィックス部のタイプ数

(7) vdc_layer_id_t

vdc_layer_id_t はレイヤ ID を表す列挙型です。

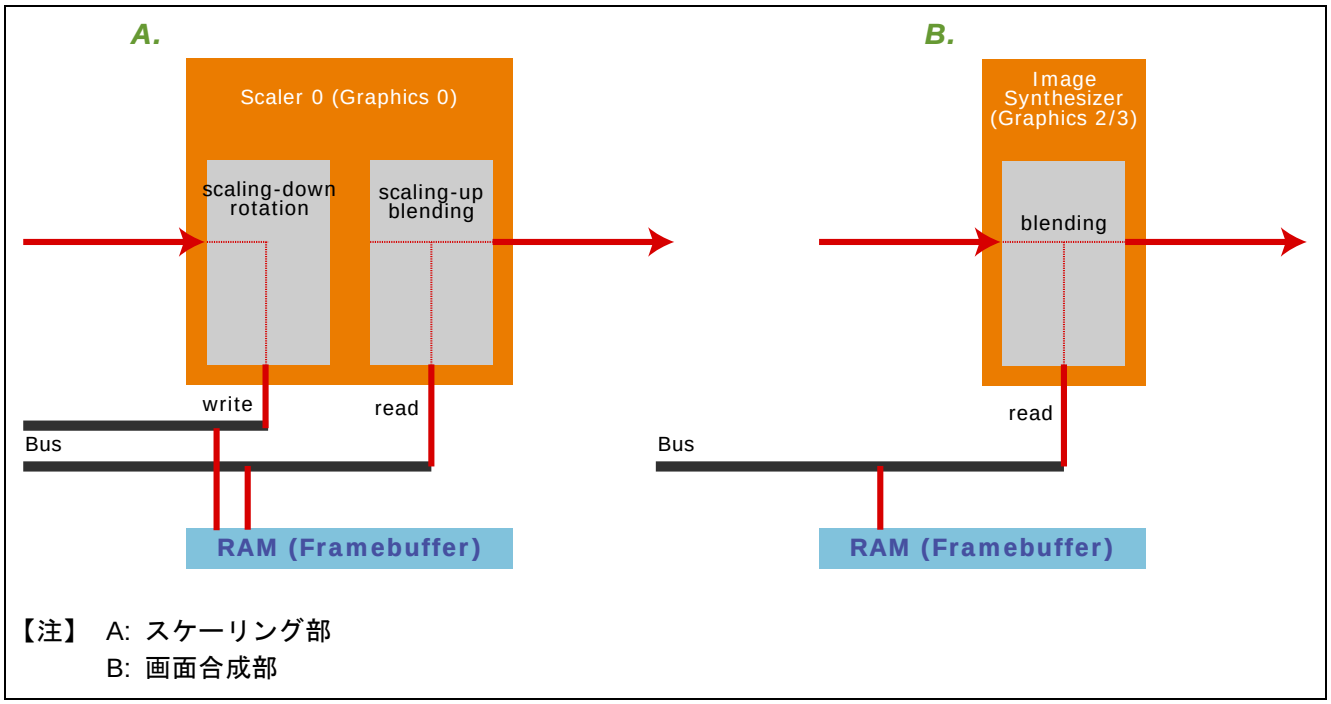
```
typedef enum
{
    VDC_LAYER_ID_ALL      = -1,
    VDC_LAYER_ID_0_WR     = (VDC_SC_TYPE_SC0 + 0),
    VDC_LAYER_ID_0_RD     = (VDC_SC_TYPE_NUM + VDC_GR_TYPE_GR0),
    VDC_LAYER_ID_2_RD     = (VDC_SC_TYPE_NUM + VDC_GR_TYPE_GR2),
    VDC_LAYER_ID_3_RD     = (VDC_SC_TYPE_NUM + VDC_GR_TYPE_GR3),
    VDC_LAYER_ID_NUM      = (VDC_SC_TYPE_NUM + VDC_GR_TYPE_NUM)
} vdc_layer_id_t;
```

列挙定数	値	説明
VDC_LAYER_ID_ALL	-1	全てのレイヤを表す ID
VDC_LAYER_ID_0_WR	0	レイヤ 0 の書き込みプロセスの ID
VDC_LAYER_ID_0_RD	1	レイヤ 0 の読み出しプロセスの ID
VDC_LAYER_ID_2_RD	2	レイヤ 2 の読み出しプロセスの ID
VDC_LAYER_ID_3_RD	3	レイヤ 3 の読み出しプロセスの ID
VDC_LAYER_ID_NUM	4	レイヤの ID 数

VDC にはグラフィックス部 0 (レイヤ 0)、グラフィックス部 2 (レイヤ 2)、そしてグラフィックス部 3 (レイヤ 3)の 3 つのレイヤがあります。VDC 内部のブロックでは、スケーリング部 0 がグラフィックス部 0 に該当し、2 つの画面合成部がそれぞれグラフィックス部 2 とグラフィックス部 3 に当たります。

スケーリング部は、入力されたデータをメモリへ書き出す前段と、メモリから読み出す後段に分けることができます(図 5-1 A 参照)。スケーリング部の前段では入力された画像データに対し、縮小処理や回転処理を行い、メモリへ書き込みます。スケーリング部の後段ではメモリから読み出したデータに対し拡大処理を行います。列挙型 vdc_layer_id_t にて定義されるレイヤ ID では、同じレイヤのメモリ書き込み処理と読み出し処理に別の ID が割り当てられています。

画面合成部では、メモリから読み出した画像データと下位レイヤの画像データを合成することができます(図 5-1 B 参照)。



【注】　A: スケーリング部
　　B: 画面合成部

図 5-1　メモリに対する書き込み／読み出し処理

レイヤ間の画像合成による結果は図 5-2 のようになります。レイヤ 0 が最下位レイヤで、レイヤ 3 が最上位レイヤとなります。

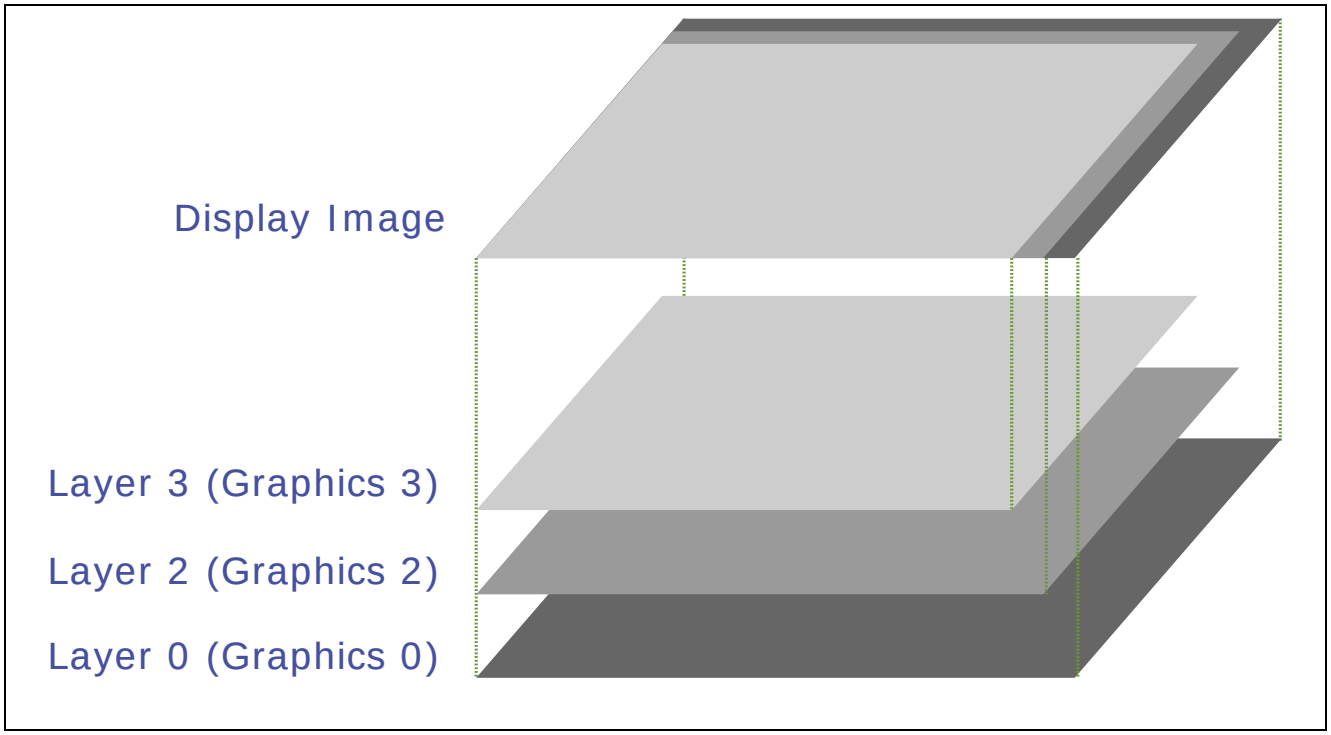


図 5-2　レイヤと画像合成

(8) vdc_int_type_t

vdc_int_type_t は VDC の割り込み種別を表す列挙型です。

```
typedef enum
{
    VDC_INT_TYPE_S0_VI_VSYNC = 0,
    VDC_INT_TYPE_S0_LO_VSYNC,
    VDC_INT_TYPE_S0_VSYNCERR,
    VDC_INT_TYPE_VLINE,
    VDC_INT_TYPE_S0_VFIELD,
    VDC_INT_TYPE_IV1_VBUFERR,
    VDC_INT_TYPE_IV3_VBUFERR,
    VDC_INT_TYPE_IV5_VBUFERR,
    VDC_INT_TYPE_IV6_VBUFERR,
    VDC_INT_TYPE_S0_WLINE,
    VDC_INT_TYPE_NUM
} vdc_int_type_t;
```

列挙定数	値	説明
VDC_INT_TYPE_S0_VI_VSYNC	0	スケーリング 0 に入力される垂直同期信号
VDC_INT_TYPE_S0_LO_VSYNC	1	スケーリング 0 から出力される垂直同期信号
VDC_INT_TYPE_S0_VSYNCERR	2	スケーリング 0 の垂直同期信号の欠落信号
VDC_INT_TYPE_VLINE	3	グラフィックス(3)パネル出力の指定ライン信号
VDC_INT_TYPE_S0_VFIELD	4	スケーリング 0 の録画機能のフィールド終了信号
VDC_INT_TYPE_IV1_VBUFERR	5	スケーリング 0 のフレームバッファ書き込みオーバフロー信号
VDC_INT_TYPE_IV3_VBUFERR	6	グラフィックス(0)フレームバッファ読み出しアンダフロー信号
VDC_INT_TYPE_IV5_VBUFERR	7	グラフィックス(2)フレームバッファ読み出しアンダフロー信号
VDC_INT_TYPE_IV6_VBUFERR	8	グラフィックス(3)フレームバッファ読み出しアンダフロー信号
VDC_INT_TYPE_S0_WLINE	9	スケーリング 0 の縮小制御部に入力される書き込み指定ライン信号
VDC_INT_TYPE_NUM	10	VDC の割り込み種別数

(9) vdc_gr_disp_sel_t

vdc_gr_disp_sel_t はグラフィックス表示設定のタイプを表す列挙型です。

```
typedef enum
{
    VDC_DISPSEL_IGNORED    = -1,
    VDC_DISPSEL_BACK       = 0,
    VDC_DISPSEL_LOWER      = 1,
    VDC_DISPSEL_CURRENT    = 2,
    VDC_DISPSEL_BLEND      = 3,
    VDC_DISPSEL_NUM        = 4
} vdc_gr_disp_sel_t;
```

列挙定数	値	説明
VDC_DISPSEL_IGNORED	-1	無視、変更なし
VDC_DISPSEL_BACK	0	背景色表示
VDC_DISPSEL_LOWER	1	下層グラフィックス表示
VDC_DISPSEL_CURRENT	2	カレントグラフィックス表示

VDC_DISPSEL_BLEND	3	下層グラフィックスとカレントグラフィックスのブレンド表示
VDC_DISPSEL_NUM	4	グラフィックス表示設定のタイプ数

(10) vdc_imgimprv_id_t

vdc_imgimprv_id_t は画質改善部を表す列挙型です。

```
typedef enum
{
    VDC_IMG_IMPRV_0 = 0,
     VDC_IMG_IMPRV_NUM
} vdc_imgimprv_id_t;
```

列挙定数	値	説明
VDC_IMG_IMPRV_0	0	画質改善部 0
VDC_IMG_IMPRV_NUM	1	画質改善部の数

(11) spea_onoff_t

spea_onoff_t は ON と OFF を表す列挙型です。SPEA の Window の ON、OFF の設定に使用します。

```
typedef enum
{
    SPEA_OFF    = 0,
    SPEA_ON     = 1
} spea_onoff_t;
```

列挙定数	値	説明
SPEA_OFF	0	OFF
SPEA_ON	1	ON

(12) rle_onoff_t

rle_onoff_t は ON と OFF を表す列挙型です。RLE 機能の ON、OFF の設定に使用します。

```
typedef enum
{
    RLE_OFF    = 0,
    RLE_ON     = 1
} rle_onoff_t;
```

列挙定数	値	説明
RLE_OFF	0	OFF
RLE_ON	1	ON

(13) spea_window_id_t

spea_window_id_t は SPEA の Window ID を表す列挙型です。SPEA のレイヤ 1 つにつき、最大 15 個の Window が作成可能です。

```
typedef enum
{
    WINDOW_00 = 0,
    WINDOW_01,
    WINDOW_02,
    WINDOW_03,
    WINDOW_04,
    WINDOW_05,
    WINDOW_06,
    WINDOW_07,
    WINDOW_08,
    WINDOW_09,
    WINDOW_10,
    WINDOW_11,
    WINDOW_12,
    WINDOW_13,
    WINDOW_14,
    WINDOW_15,
    WINDOW_NUM
} spea_window_id_t;
```

列挙定数	値	説明
WINDOW_00	0	SPEA の Window ID 最下位
WINDOW_01	1	SPEA の Window ID
WINDOW_02	2	SPEA の Window ID
WINDOW_03	3	SPEA の Window ID
WINDOW_04	4	SPEA の Window ID
WINDOW_05	5	SPEA の Window ID
WINDOW_06	6	SPEA の Window ID
WINDOW_07	7	SPEA の Window ID
WINDOW_08	8	SPEA の Window ID
WINDOW_09	9	SPEA の Window ID
WINDOW_10	10	SPEA の Window ID
WINDOW_11	11	SPEA の Window ID
WINDOW_12	12	SPEA の Window ID
WINDOW_13	13	SPEA の Window ID
WINDOW_14	14	SPEA の Window ID
WINDOW_15	15	SPEA の Window ID 最上位
WINDOW_NUM	16	SPEA の Window ID 数

5.3　共通構造体定義

(1)　vdc_period_rect_t

vdc_period_rect_t は VDC の信号の水平／垂直タイミングを表す構造体です。

```
typedef struct
{
    uint16_t    vs;
    uint16_t    vw;
    uint16_t    hs;
    uint16_t    hw;
} vdc_period_rect_t;
```

型 メンバ名	説明
uint16_t vs	基準信号からの垂直信号開始位置 (ライン数)
uint16_t vw	垂直信号幅 (ライン数)
uint16_t hs	基準信号からの水平信号開始位置 (クロック数)
uint16_t hw	水平信号幅 (クロック数)

vdc_period_rect_t 構造体の水平／垂直タイミングは図 5-3 に示されるように、矩形の領域として表現されます。

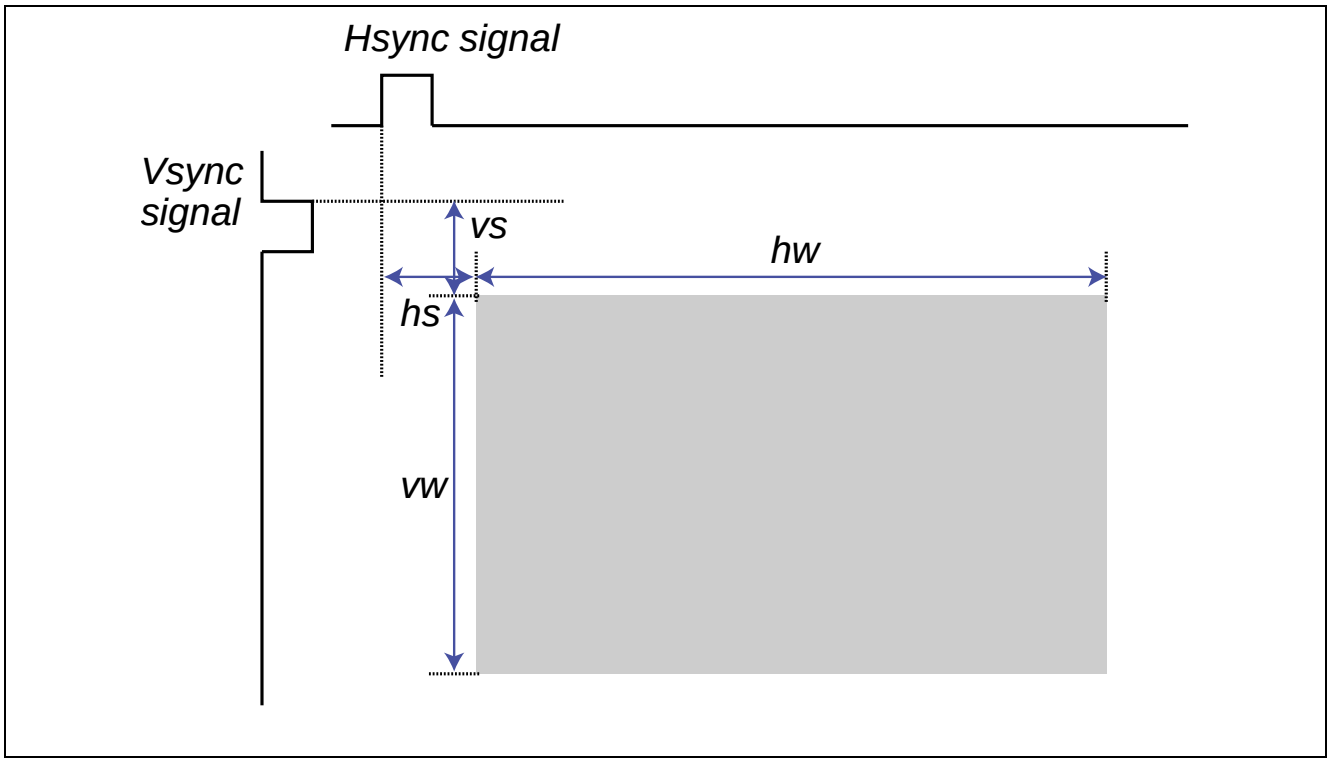


図 5-3　水平／垂直タイミングを表す矩形

5.4　エラーコード

VDC ドライバのエラーコードの一覧を表 5-4 に示します。

表 5-4　VDC ドライバのエラーコード一覧

エラーコード	値	説明 (エラータイプ)
VDC_OK	0	正常終了
VDC_ERR_PARAM_CHANNEL	1	チャネル不正エラー (パラメータエラー) 許可されていないチャネルが指定されました。
VDC_ERR_PARAM_LAYER_ID	2	レイヤ ID 不正エラー (パラメータエラー) 許可されていないレイヤ ID が指定されました。
VDC_ERR_PARAM_NULL	3	NULL 指定エラー (パラメータエラー) 指定すべきパラメータに対し、NULL が指定されました。
VDC_ERR_PARAM_BIT_WIDTH	4	ビット幅エラー (パラメータエラー) 設定可能なビット幅を越えた値が指定されました。
VDC_ERR_PARAM_UNDEFINED	5	未定義パラメータ指定エラー (パラメータエラー) 仕様に定義されていない値が指定されました。
VDC_ERR_PARAM_EXCEED_RANGE	6	設定範囲外エラー (パラメータエラー) 指定されたパラメータの値は仕様に定義された範囲を越えています。
VDC_ERR_PARAM_CONDITION	7	不許可条件エラー (パラメータエラー) 仕様により許可されていない条件において、パラメータが指定されました。
VDC_ERR_IF_CONDITION	8	インタフェース条件エラー (インタフェースエラー) 許可されていない条件において API 関数が呼び出されました。
VDC_ERR_RESOURCE_CLK	9	クロックリソースエラー (リソースエラー) パネルクロックが設定されていません。
VDC_ERR_RESOURCE_VSYNC	10	垂直同期信号リソースエラー (リソースエラー) 垂直同期信号が設定されていません。
VDC_ERR_RESOURCE_INPUT	11	入力信号リソースエラー (リソースエラー) ビデオ映像入力が設定されていません。
VDC_ERR_RESOURCE_OUTPUT	12	出力リソースエラー (リソースエラー) ディスプレイ出力が設定されていません。
VDC_ERR_RESOURCE_LVDS_CLK	13	LVDS クロックリソースエラー（リソースエラー） LVDS クロックが設定されていません。
VDC_ERR_RESOURCE_LAYER	14	レイヤリソースエラー (リソースエラー) 指定されたレイヤが利用できない条件において指定されました。
VDC_ERR_NUM	15	VDC ドライバのエラーコード数

SPEA ドライバのエラーコードの一覧を表 5-5 に示します。

表 5-5　SPEA ドライバのエラーコード一覧

エラーコード	値	説明 (エラータイプ)
SPEA_OK	0	正常終了
SPEA_ERR_PARAM_LAYER_ID	1	レイヤ ID 不正エラー (パラメータエラー) 許可されていないレイヤ ID が指定されました。
SPEA_ERR_PARAM_NULL	2	NULL 指定エラー (パラメータエラー) 指定すべきパラメータに対し、NULL が指定されました。
SPEA_ERR_PARAM	3	不許可条件エラー (パラメータエラー) 仕様により許可されていない条件において、パラメータが指定されました。

5.5　ユーザカスタムパラメータ

本ドライバでは"r_vdc_user.h"において、ユーザが静的に変更可能なパラメータが定義されています。

(1)　列挙型 vdc_colcnv_rgb_ycbcr_t

vdc_colcnv_rgb_ycbcr_t はカラーマトリクスの設定値を表す列挙型です。GBR 信号を YCbCr 信号へ変換する時に、VDC ドライバにより参照されます。デフォルトの設定値は、ハードウェアマニュアルに記載されている、標準値です。

```
typedef enum
{
    VDC_COLORCONV_Y_R  = (77u),
    VDC_COLORCONV_Y_G  = (150u),
    VDC_COLORCONV_Y_B  = (29u),
    VDC_COLORCONV_CB_R = (2005u),
    VDC_COLORCONV_CB_G = (1963u),
    VDC_COLORCONV_CB_B = (128u),
    VDC_COLORCONV_CR_R = (128u),
    VDC_COLORCONV_CR_G = (1941u),
    VDC_COLORCONV_CR_B = (2027u)
} vdc_colcnv_rgb_ycbcr_t;
```

列挙定数	値	説明
VDC_COLORCONV_Y_R	77u	Y/G 信号出力の Cr/R 信号ゲイン調整 (0.299)
VDC_COLORCONV_Y_G	150u	Y/G 信号出力の Y/G 信号ゲイン調整 (0.587)
VDC_COLORCONV_Y_B	29u	Y/G 信号出力の Cb/B 信号ゲイン調整 (0.114)
VDC_COLORCONV_CB_R	2005u	Cb/B 信号出力の Cr/R 信号ゲイン調整 (-0.169)
VDC_COLORCONV_CB_G	1963u	Cb/B 信号出力の Y/G 信号ゲイン調整 (-0.331)
VDC_COLORCONV_CB_B	128u	Cb/B 信号出力の Cb/B 信号ゲイン調整 (0.500)
VDC_COLORCONV_CR_R	128u	Cr/R 信号出力の Cr/R 信号ゲイン調整 (0.500)
VDC_COLORCONV_CR_G	1941u	Cr/R 信号出力の Y/G 信号ゲイン調整 (-0.419)
VDC_COLORCONV_CR_B	2027u	Cr/R 信号出力の Cb/B 信号ゲイン調整 (-0.081)

【注】 値は 11 ビットの 2 の補数で表現されます。
(-1024 ~ +1023[LSB]、256[LSB] = 1.0[倍])

(2)　列挙型 vdc_colcnv_ycbcr_rgb_t

vdc_colcnv_ycbcr_rgb_t はカラーマトリクスの設定値を表す列挙型です。YCbCr 信号を GBR 信号へ変換する時に、VDC ドライバにより参照されます。デフォルトの設定値は、ハードウェアマニュアルに記載されている、標準値です。

```
typedef enum
{
    VDC_COLORCONV_G_Y  = (256u),
    VDC_COLORCONV_G_CB = (1960u),
    VDC_COLORCONV_G_CR = (1865u),
    VDC_COLORCONV_B_Y  = (256u),
    VDC_COLORCONV_B_CB = (454u),
    VDC_COLORCONV_B_CR = (0u),
    VDC_COLORCONV_R_Y  = (256u),
    VDC_COLORCONV_R_CB = (0u),
    VDC_COLORCONV_R_CR = (359u)
} vdc_colcnv_ycbcr_rgb_t;
```

列挙定数	値	説明
VDC_COLORCONV_G_Y	256u	Y/G 信号出力の Y/G 信号ゲイン調整 (1.000)
VDC_COLORCONV_G_CB	1960u	Y/G 信号出力の Cb/B 信号ゲイン調整 (-0.344)
VDC_COLORCONV_G_CR	1865u	Y/G 信号出力の Cr/R 信号ゲイン調整 (-0.714)
VDC_COLORCONV_B_Y	256u	Cb/B 信号出力の Y/G 信号ゲイン調整 (1.000)
VDC_COLORCONV_B_CB	454u	Cb/B 信号出力の Cb/B 信号ゲイン調整 (1.772)
VDC_COLORCONV_B_CR	0u	Cb/B 信号出力の Cr/R 信号ゲイン調整 (0.000)
VDC_COLORCONV_R_Y	256u	Cr/R 信号出力の Y/G 信号ゲイン調整 (1.000)
VDC_COLORCONV_R_CB	0u	Cr/R 信号出力の Cb/B 信号ゲイン調整 (0.000)
VDC_COLORCONV_R_CR	359u	Cr/R 信号出力の Cr/R 信号ゲイン調整 (1.402)

【注】 値は 11 ビットの 2 の補数で表現されます。
(-1024 ~ +1023[LSB]、256[LSB] = 1.0[倍])

(3)　定数定義

以下に定数を記載します。

定数	値	説明
VDC_COLORCONV_DC_OFFSET	128u	カラーマトリクスの Y/G、B、R 信号のオフセット(DC)調整値 符号無し(0 ~ 255、128[LSB] = 0) カラーマトリクス設定時に VDC ドライバにより参照されます。
VDC_COLORCONV_1TIMES_GAIN	256u	カラーマトリクスの 1.0[倍]ゲイン設定値 -1024 ~ +1023[LSB]、256[LSB] = 1.0[倍] YCbCr 信号を YCbCr 信号へ変換する時、及び GBR 信号を GBR 信号へ変換する時に VDC ドライバにより参照されます。
VDC_GAM_GAIN_ADJ_NUM	32u	ガンマ補正の各信号のゲイン設定数
VDC_GAM_START_TH_NUM	31u	ガンマ補正の各信号の領域開始閾値設定数

5.6　コンパイルスイッチ

本ドライバでは"r_vdc_user.h"において以下のコンパイルスイッチが定義されています。

表 5-6　コンパイルスイッチ

コンパイルスイッチ	説明
R_VDC_CHECK_PARAMETERS	この定義を有効にすると、VDC ドライバ API 関数の呼び出し時に引数のパラメータチェックを行います。パラメータチェックの結果、エラーがある場合はパラメータエラーを表すエラーコードを返します。エラーコードについては「5.4 エラーコード」を参照してください。

5.7　制限事項

(1)　予約語

本ドライバでは他のプログラムと区別する為、関数や変数名などのシンボルに以下のプレフィックスを付加しています。大文字、小文字を問わず以下に示されたシンボルから始まる名称は使用しないでください。

- R_VDC
- VDC
- R_SPEA
- SPEA

(2)　レジスタ更新

VDC や SPEA では多くのレジスタが、垂直同期信号の立ち上がりのタイミングで設定の更新を反映します。従って、値が設定されてから反映されるまでに最大で垂直同期信号 1 周期分の時間が掛かります。

(3)　再入可能性

本ドライバの API は再入可能ではありません。本ドライバの API を複数のタスクや割り込み処理から非同期に呼び出した場合、予期せぬ動作をする可能性があります。ドライバコールの呼び出し元やタイミングについては注意してください。

(4)　レジスタへのアクセス

このドライバは、VDC や SEPA の全てのレジスタに対してアクセスする手段を提供しません。また、一部のレジスタについてはドライバ側で自動的に設定します。

5.8　VDC 内部のモジュール構成

VDC 内部のモジュール構成とデータの流れについて、図 5-4 に示します。

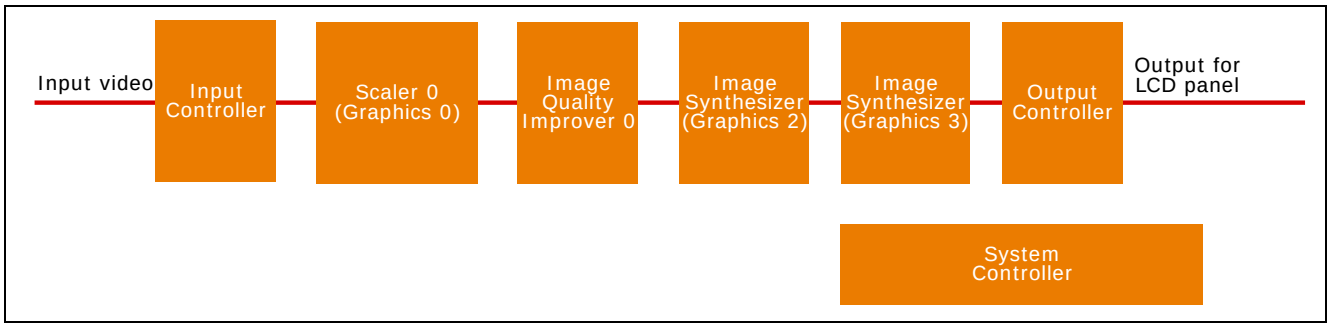


図 5-4　VDC 内部の構成

RZ/A2M に搭載されている VDC は 6 つのブロックから構成されています。

1. Input Controller: 入力制御部
   同期調整、入力映像信号に対する調整
2. Scaler 0: スケーリング部 0 (グラフィックス部 0)
   入力映像のスケーリング＆回転
3. Image Quality Improver 0: 画質改善部 0
   画質調整、カラーマトリクス機能による色変換
4. Image Synthesizer: 画面合成部 (グラフィックス部 2、グラフィックス部 3)
   画像を重ね合わせる機能
5. Output Controller: 出力制御部
   出力画像調整、出力フォーマット変換、TFT-LCD パネル向けの制御信号出力機能
6. System Controller: システム制御部
   割り込み制御、パネルクロック制御

6. 関数リファレンス

VDC ドライバの API 関数の一覧を表 6-1 に示します。

表 6-1　API 関数一覧

関数名	項番	概要
R_VDC_Initialize	6.1	VDC ドライバ初期化処理
R_VDC_Terminate	6.2	VDC ドライバ終了処理
R_VDC_VideoInput	6.3	ビデオ入力設定処理
R_VDC_SyncControl	6.4	同期制御設定処理
R_VDC_DisplayOutput	6.5	ディスプレイ出力設定処理
R_VDC_CallbackISR	6.6	割り込みコールバック設定処理
R_VDC_WriteDataControl	6.7	データ書き込み制御処理
R_VDC_ChangeWriteProcess	6.8	データ書き込み変更処理
R_VDC_ReadDataControl	6.9	データ読み出し制御処理
R_VDC_ChangeReadProcess	6.10	データ読み出し変更処理
R_VDC_StartProcess	6.11	データ書き込み／読み出し開始処理
R_VDC_StopProcess	6.12	データ書き込み／読み出し停止処理
R_VDC_ReleaseDataControl	6.13	データ書き込み／読み出し制御解放処理
R_VDC_VideoNoiseReduction	6.14	ノイズリダクション設定処理
R_VDC_ImageColorMatrix	6.15	カラーマトリクス設定処理
R_VDC_ImageEnhancement	6.16	画質改善設定処理
R_VDC_ImageBlackStretch	6.17	黒伸張設定処理
R_VDC_AlphaBlending	6.18	アルファブレンディング設定処理
R_VDC_AlphaBlendingRect	6.19	矩形領域アルファブレンディング設定処理
R_VDC_Chromakey	6.20	クロマキー設定処理
R_VDC_CLUT	6.21	CLUT 設定処理
R_VDC_DisplayCalibration	6.22	画面出力較正処理
R_VDC_GammaCorrection	6.23	ガンマ補正設定処理
R_VDC_GetISR	6.24	割り込みサービスルーチン取得処理

SPEA ドライバの API 関数の一覧を表 6-2 に示します。

表 6-2　API 関数一覧

関数名	項番	概要
R_SPEA_WindowOffset	6.25	Window の座標オフセットの設定
R_SPEA_SetWindow	6.26	Window パラメータの設定
R_SPEA_WindowUpdate	6.27	Window パラメータの更新要求
R_RLE_SetWindow	6.28	RLE パラメータの設定
R_RLE_WindowUpdate	6.29	RLE パラメータの更新要求

6.1 R_VDC_Initialize

概要	VDC ドライバ初期化処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_Initialize(const vdc_channel_t ch, const vdc_init_t * const param, void (* const init_func)(uint32_t), const uint32_t user_num);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_init_t * param: 初期化処理パラメータ • void (* init_func)(uint32_t): ユーザ定義関数のポインタ VDC ドライバ初期化処理に合わせて実行すべき処理をユーザが実装して指定します。API 関数 R_VDC_Initialize 内では、VDC のレジスタ設定に先駆けてこの関数が呼び出されます。呼び出し時には user_num が引数となります。 不要な場合は'0'を指定してください。 • uint32_t user_num: ユーザ定義番号 ユーザ定義関数 init_func へ渡す引数を指定します。ユーザ定義関数に'0'が指定された場合、本パラメータは無視されます。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー

詳細

(1) 機能

本関数では、VDC ドライバの初期化とそれに伴う以下の処理を行います。

- VDC ドライバの内部変数の初期化
- init_func で指定されたユーザ定義関数の呼び出し
- VDC のパネルクロックの設定、及びイネーブル設定
- VDC の全ての割り込みのディセーブル設定

(2) 使用条件

VDC ドライバの動作には、事前に以下の処理が実行されている必要があります。

- VDC モジュールへのクロック供給
- VDC に関係する割り込みの設定 (割り込みサービスルーチン、割り込み優先度)
- VDC に関係する I/O ポートの設定
- LCD パネルやビデオ入力に必要な環境固有の設定

LCD パネル出力に関係する I/O ポートの設定以外の上記処理は、本関数の呼び出し前に実行するか、上記処理を実施する関数をユーザが実装し、ユーザ定義関数として init_func へ指定してください。

(3) パラメータ詳細

vdc_init_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_panel_clksel_t    panel_icksel;
    vdc_panel_clk_dcdr_t  panel_dcdr;
    const vdc_lvds_t    * lvds;
} vdc_init_t;
```

型 メンバ名	説明
vdc_panel_clksel_t panel_icksel	パネルクロック選択 • VDC_PANEL_ICKSEL_IMG_DV: 映像クロック (DV_CLK)の分周クロック • VDC_PANEL_ICKSEL_EXT_0: 外部クロック 0 (LCD0_EXTCLK)の分周クロック • VDC_PANEL_ICKSEL_PERI: 周辺クロック 1 (P1φ)の分周クロック • VDC_PANEL_ICKSEL_LVDS: LVDS PLL のクロック • VDC_PANEL_ICKSEL_LVDS_DIV7: LVDS PLL の 7 分周クロック
vdc_panel_clk_dcdr_t panel_dcdr	クロック分周比設定 • VDC_PANEL_CLKDIV_1_1: 1/1 • VDC_PANEL_CLKDIV_1_2: 1/2 • VDC_PANEL_CLKDIV_1_3: 1/3 • VDC_PANEL_CLKDIV_1_4: 1/4 • VDC_PANEL_CLKDIV_1_5: 1/5 • VDC_PANEL_CLKDIV_1_6: 1/6 • VDC_PANEL_CLKDIV_1_7: 1/7 • VDC_PANEL_CLKDIV_1_8: 1/8 • VDC_PANEL_CLKDIV_1_9: 1/9 • VDC_PANEL_CLKDIV_1_12: 1/12 • VDC_PANEL_CLKDIV_1_16: 1/16 • VDC_PANEL_CLKDIV_1_24: 1/24 • VDC_PANEL_CLKDIV_1_32: 1/32
const vdc_lvds_t * lvds	LVDS 関連パラメータ 不要な場合は NULL を設定してください。

vdc_lvds_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_lvds_in_clk_sel_t   lvds_in_clk_sel;
    vdc_lvds_ndiv_t         lvds_idiv_set;    /* Not use */
    uint16_t                lvdspll_tst;      /* Not use */
    vdc_lvds_ndiv_t         lvds_odiv_set;
    vdc_channel_t           lvds_vdc_sel;
    uint16_t                lvdspll_fd;
    uint16_t                lvdspll_rd;
    vdc_lvds_pll_nod_t      lvdspll_od;       /* Not use */
} vdc_lvds_t;
```

型 メンバ名	説明
vdc_lvds_in_clk_sel_t lvds_in_clk_sel	分周器 1 への入力クロック選択 • VDC_LVDS_INCLK_SEL_DV_0: DV0_CLK0 • VDC_LVDS_INCLK_SEL_EXT_0: LCD0_EXTCLK • VDC_LVDS_INCLK_SEL_PERI: P1φ
vdc_lvds_ndiv_t lvds_idiv_set	分周器 1 の分周数 NIDIV 設定(未使用) • VDC_LVDS_NDIV_1: NIDIV = 1 • VDC_LVDS_NDIV_2: NIDIV = 2 • VDC_LVDS_NDIV_4: NIDIV = 4
uint16_t lvdspll_tst	LVDS PLL の内部パラメータ設定(未使用)
vdc_lvds_ndiv_t lvds_odiv_se	分周器 2 の分周数 NODIV 設定 • VDC_LVDS_NDIV_1: NODIV = 1 • VDC_LVDS_NDIV_2: NODIV = 2 • VDC_LVDS_NDIV_4: NODIV = 4
vdc_channel_t lvds_vdc_sel	LVDS から出力する VDC のチャネル選択 • VDC_CHANNEL_0
uint16_t lvdspll_fd	LVDS PLL の帰還分周 NFD 設定 NFD = lvdspll_fd + 1 lvdspll_fd (22 ~ 62)
uint16_t lvdspll_rd	LVDS PLL の入力分周 NRD 設定 NRD = lvdspll_rd + 1 lvdspll_rd (0 ~ 7)
vdc_lvds_pll_nod_t lvdspll_od	LVDS PLL の出力分周 NOD 設定(未使用) • VDC_LVDS_PLL_NOD_1: NOD = 1 • VDC_LVDS_PLL_NOD_2: NOD = 2 • VDC_LVDS_PLL_NOD_4: NOD = 4 • VDC_LVDS_PLL_NOD_8: NOD = 8

6.2 R_VDC_Terminate

概要	VDC ドライバ終了処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_Terminate(const vdc_channel_t ch, void (* const quit_func)(uint32_t), const uint32_t user_num);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • void (* quit_func)(uint32_t): ユーザ定義関数のポインタ VDC ドライバ終了処理に合わせて実行すべき処理をユーザが実装して指定します。API 関数 R_VDC_Terminate 内では、VDC のレジスタ設定の後にこの関数が呼び出されます。呼び出し時には user_num が引数となります。 不要な場合は'0'を指定してください。 • uint32_t user_num: ユーザ定義番号 ユーザ定義関数 quit_func へ渡す引数を指定します。ユーザ定義関数に'0'が指定された場合、本パラメータは無視されます。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー

詳細

(1) 機能

本関数では、VDC ドライバの終了とそれに伴う以下の処理を行います。

- VDC の全ての割り込みのディセーブル設定
- VDC のパネルクロックのディセーブル設定
- quit_func で指定されたユーザ定義関数の呼び出し

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

6.3 R_VDC_VideoInput

概要	ビデオ入力設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_VideoInput(const vdc_channel_t ch, const vdc_input_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_input_t * param: ビデオ入力設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_PARAM_CONDITION: 不許可条件エラー

詳細

(1) 機能

本関数では、ビデオ入力に関する以下の処理を行います。

- 入力信号の位相タイミング設定
- ビデオ入力に伴う同期信号の遅延制御
- 外部入力ビデオ信号のパラメータ設定

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

(3) パラメータ詳細

vdc_input_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_input_sel_t          inp_sel;
    uint16_t                 inp_fh50;
    uint16_t                 inp_fh25;
    const vdc_sync_delay_t   * dly;
    const vdc_ext_in_sig_t   * ext_sig;
} vdc_input_t;
```

型 メンバ名	説明
vdc_input_sel_t inp_sel	入力選択 • VDC_INPUT_SEL_EXT (1): 外部入力端子 必ず VDC_INPUT_SEL_EXT を設定してください。
uint16_t inp_fh50	垂直同期の 1/2fH 位相タイミング設定 0x0000 ~ 0x03FF 必ず水平周期の 1/2 クロック周期を設定してください。
uint16_t inp_fh25	垂直同期の 1/4fH 位相タイミング設定 0x0000 ~ 0x03FF 必ず水平周期の 1/4 クロック周期を設定してください。
const vdc_sync_delay_t * dly	同期遅延調整 NULL が指定された場合、設定は変更されません。ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については構造体の説明を参照ください。
const vdc_ext_in_sig_t * ext_sig	外部入力信号パラメータ NULL は設定しないでください。

vdc_sync_delay_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t    inp_vs_dly_l;
    uint16_t    inp_fld_dly;
    uint16_t    inp_vs_dly;
    uint16_t    inp_hs_dly;
} vdc_sync_delay_t;
```

型 メンバ名	初期値	説明
uint16_t inp_vs_dly_l	0	垂直同期信号、フィールド判別のライン遅延量 0 ~ 7 [ライン]
uint16_t inp_fld_dly	0	フィールド判別信号の遅延量 0 ~ 254 [クロック]
uint16_t inp_vs_dly	0	垂直同期信号の遅延量 0 ~ 254 [クロック]
uint16_t inp_hs_dly	0	水平同期信号の遅延量 0 ~ 254 [クロック]

vdc_ext_in_sig_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_extin_format_t    inp_format;
    vdc_edge_t           inp_pxd_edge;
    vdc_edge_t           inp_vs_edge;
    vdc_edge_t           inp_hs_edge;
    vdc_onoff_t          inp_endian_on;
    vdc_onoff_t          inp_swap_on;
    vdc_sig_pol_t        inp_vs_inv;
    vdc_sig_pol_t        inp_hs_inv;
    vdc_extin_ref_hsync_t  inp_h_edge_sel;
    vdc_extin_input_line_t inp_f525_625;
    vdc_extin_h_pos_t     inp_h_pos;
} vdc_ext_in_sig_t;
```

型 メンバ名	説明
vdc_extin_format_t inp_format	外部入力のフォーマット選択 • VDC_EXTIN_FORMAT_RGB888 (0): RGB888 • VDC_EXTIN_FORMAT_RGB666 (1): RGB666 • VDC_EXTIN_FORMAT_RGB565 (2): RGB565 • VDC_EXTIN_FORMAT_BT656 (3): BT6556 • VDC_EXTIN_FORMAT_BT601 (4): BT6501 • VDC_EXTIN_FORMAT_YCBCR422 (5): YCbCr422 • VDC_EXTIN_FORMAT_YCBCR444 (6): YCbCr444
vdc_edge_t inp_pxd_edge	外部入力の映像信号 DV_DATA の入力段取り込みクロックのエッジ選択 • VDC_EDGE_RISING: 立ち上がりエッジ • VDC_EDGE_FALLING: 立ち下りエッジ
vdc_edge_t inp_vs_edge	外部入力の垂直同期信号 DV_VSYNC の入力段取り込みクロックのエッジ選択 • VDC_EDGE_RISING: 立ち上がりエッジ • VDC_EDGE_FALLING: 立ち下りエッジ
vdc_edge_t inp_hs_edge	外部入力の水平同期信号 DV_HSYNC の入力段取り込みクロックのエッジ選択 • VDC_EDGE_RISING: 立ち上がりエッジ • VDC_EDGE_FALLING: 立ち下りエッジ
vdc_onoff_t inp_endian_on	外部入力のビットエンディアン変更 • VDC_OFF • VDC_ON
vdc_onoff_t inp_swap_on	外部入力の B/R 信号入れ替え • VDC_OFF • VDC_ON
vdc_sig_pol_t inp_vs_inv	外部入力の垂直同期信号 DV_VSYNC の反転制御 • VDC_SIG_POL_NOT_INVERTED: 非反転 (正極性) • VDC_SIG_POL_INVERTED: 反転 (負極性)
vdc_sig_pol_t inp_hs_inv	外部入力の水平同期信号 DV_HSYNC の反転制御 • VDC_SIG_POL_NOT_INVERTED: 非反転 (正極性) • VDC_SIG_POL_INVERTED: 反転 (負極性)
vdc_extin_ref_hsync_t	外部入力系統の BT656 水平同期信号の基準選択

inp_h_edge_sel	• VDC_EXTIN_REF_H_EAV (0): EAV 基準 • VDC_EXTIN_REF_H_SAV (1): SAV 基準
vdc_extin_input_line_t inp_f525_625	外部入力系統の BT656 入力時のライン数設定 • VDC_EXTIN_LINE_525 (0): 525 ライン • VDC_EXTIN_LINE_625 (1): 625 ライン
vdc_extin_h_pos_t inp_h_pos	水平同期基準に対するデータ列の開始タイミング設定 • VDC_EXTIN_H_POS_CBYCRY (0): Cb/Y/Cr/Y (BT656/601)、Cb/Cr (YCbCr422) • VDC_EXTIN_H_POS_YCRYCB (1): Y/Cr/Y/Cb (BT656/601)、設定禁止 (YCbCr422) • VDC_EXTIN_H_POS_CRYCBY (2): Cr/Y/Cb/Y (BT656/601)、設定禁止 (YCbCr422) • VDC_EXTIN_H_POS_YCBYCR (3): Y/Cb/Y/Cr (BT656/601)、Cr/Cb (YCbCr422)

外部入力のフォーマット選択 (inp_format)で YCbCr422 を選択した場合、水平同期基準に対するデータ列の開始タイミング設定 (inp_h_pos)に VDC_EXTIN_H_POS_YCRYCB や VDC_EXTIN_H_POS_CRYCBY を指定すると、不許可条件エラー (VDC_ERR_PARAM_CONDITION)を返します。

6.4 R_VDC_SyncControl

概要	同期制御設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_SyncControl(const vdc_channel_t ch, const vdc_sync_ctrl_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_sync_ctrl_t * param: 同期制御設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_CLK: クロックリソースエラー — VDC_ERR_RESOURCE_INPUT: 入力信号リソースエラー

詳細

(1) 機能

本関数では、同期制御に関する以下の処理を行います。

- 垂直同期信号の選択
- 同期信号の周期設定
- 垂直同期信号の遅延設定
- フル画面イネーブルの設定
- 垂直同期信号補償設定

本関数による設定は、ハードウェアのリセットか、本関数による別の設定で上書きされるまでは有効です。

(2) 使用条件

本関数の使用時には、パネルクロックが設定されている必要があります。パネルクロックが設定されていない場合、クロックリソースエラー (VDC_ERR_RESOURCE_CLK)を返します。

また、本関数で指定する垂直同期信号として外部入力垂直同期信号を選択する場合、本関数の使用前に関数「R_VDC_VideoInput」を呼び出すことで、ビデオ入力を有効にする必要があります。ビデオ入力が無効な場合、入力信号リソースエラー (VDC_ERR_RESOURCE_INPUT)を返します。

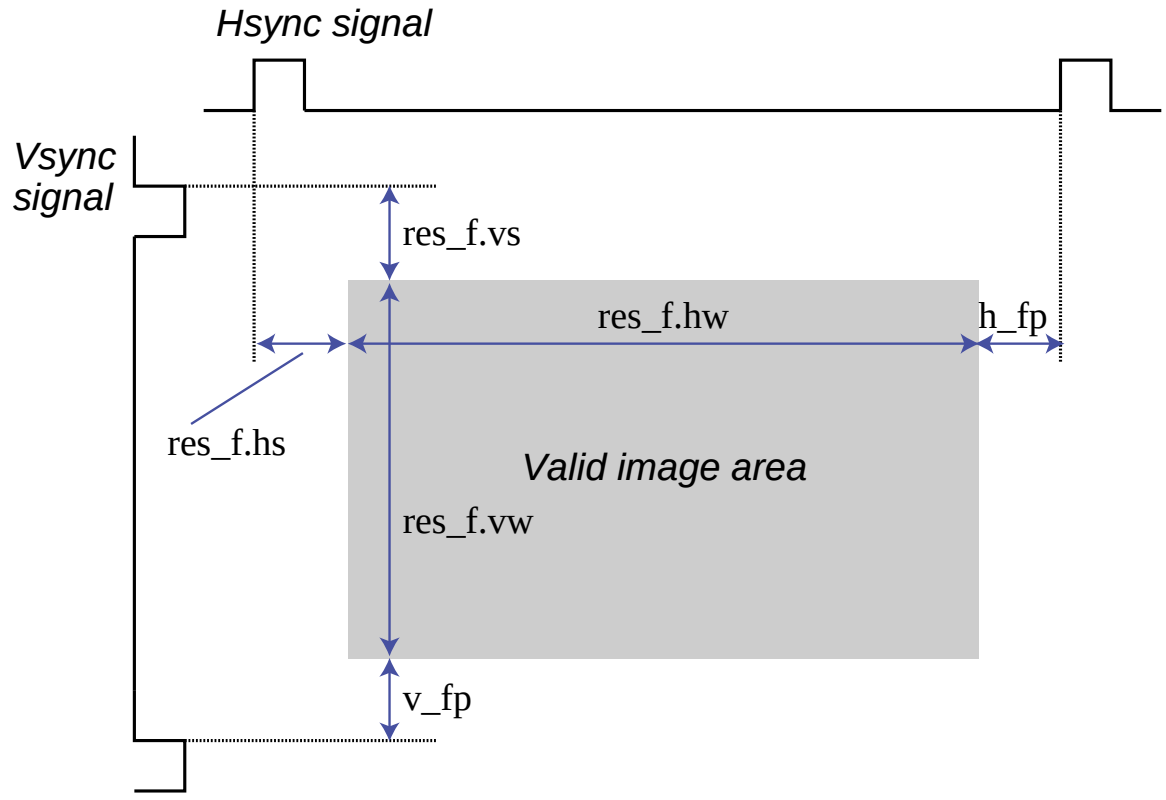
(3) パラメータ詳細

vdc_sync_ctrl_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_onoff_t              res_vs_sel;
    vdc_res_vs_in_sel_t       res_vs_in_sel;
    uint16_t                 res_fv;
    uint16_t                 res_fh;
    uint16_t                 res_vsdly;
    vdc_period_rect_t         res_f;
    const vdc_vsync_cpmpe_t  * vsync_cpmpe;
} vdc_sync_ctrl_t;
```

型 メンバ名	説明
vdc_onoff_t res_vs_sel	出力する垂直同期信号の選択 (自走同期信号) • VDC_OFF: 外部入力垂直同期信号 • VDC_ON: 内部生成した自走用垂直同期信号
vdc_res_vs_in_sel_t res_vs_in_sel	出力する水平/垂直同期、フル画面イネーブル信号の選択 • VDC_RES_VS_IN_SEL_SC0 (0): スケーリング部 0 出力 必ず VDC_RES_VS_IN_SEL_SC0 を設定してください。
uint16_t res_fv	自走用垂直同期信号の周期設定 自走用垂直同期信号周期 = (res_fv + 1) x 水平周期[usec] 0x0000 ~ 0x07FF
uint16_t res_fh	水平同期信号の周期設定 水平同期信号周期[usec] = (res_fh + 1) / ピクセルクロック[MHz] 0x0000 ~ 0x07FF
uint16_t res_vsdly	垂直同期信号遅延制御 垂直同期信号を出力水平周期単位にて遅延調整 0 ~ 255
vdc_period_rect_t res_f	フル画面イネーブル vdc_period_rect_t 構造体については「5.3(1)」を参照ください。 res_f.vs は 4 ライン以上、(res_f.vs + res_f.vw)は 2039 ライン以内になるように設定してください。 res_f.hs は 16 クロック以上、(res_f.hs + res_f.hw)は 2015 クロック以内になるように設定してください。 また、本設定については図 6-1 とその説明も参照ください。
const vdc_vsync_cpmpe_t * vsync_cpmpe	垂直同期信号補償 NULL が指定された場合、垂直同期信号の多発マスクと欠落補償は OFF となります。 欠落補償が OFF の時、欠落補償期間はドライバにより最大値 (0xFFFF)が設定されます。

フル画面イネーブルの設定と出力画像の有効期間について、図 6-1 に示します。水平同期信号の前後 16 クロックと垂直同期信号の前後 4 ラインは画像の有効期間になりません。



【注】 res_f.vs: フル画面の垂直イネーブル信号開始位置 (4 ライン以上)
res_f.vw: フル画面の垂直イネーブル信号幅
v_fp: フル画面の垂直イネーブル信号終了から垂直同期信号までの期間 (4 ライン以上)
res_f.hs: フル画面の水平イネーブル信号開始位置 (16 クロック以上)
res_f.hw: フル画面の水平イネーブル信号幅
h_fp: フル画面の水平イネーブル信号終了から水平同期信号までの期間 (16 クロック以上)

図 6-1　フル画面イネーブルの設定

vdc_vsync_cpmpe_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t       res_vmask;
    uint16_t       res_vlack;
} vdc_vsync_cpmpe_t;
```

型 メンバ名	説明
uint16_t res_vmask	垂直同期信号の多発マスク期間設定 マスク期間[usec] = res_vmask x 128 / ピクセルクロック [MHz]
uint16_t res_vlack	垂直同期信号の欠落補償期間設定 ウェイト期間[usec] = res_vlack x 128 / ピクセルクロック [MHz]

6.5 R_VDC_DisplayOutput

概要	ディスプレイ出力設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_DisplayOutput(const vdc_channel_t ch, const vdc_output_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_output_t * param: ディスプレイ出力設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_CONDITION: 不許可条件エラー — VDC_ERR_RESOURCE_CLK: クロックリソースエラー — VDC_ERR_RESOURCE_VSYNC: 垂直同期信号リソースエラー

詳細

(1) 機能

本関数では、ディスプレイ出力に関する以下の処理を行います。

- LCD パネル駆動用タイミング信号の設定
- LCD パネル出力データの位相、データ順序、フォーマットの設定
- 背景色の設定

本関数による設定は、ハードウェアのリセットか、本関数による別の設定で上書きされるまでは有効です。

(2) 使用条件

本関数の使用時には、パネルクロックと同期信号が設定されている必要があります。パネルクロックが設定されていない場合、クロックリソースエラー (VDC_ERR_RESOURCE_CLK)を返し、同期信号が設定されていない場合、垂直同期信号リソースエラー (VDC_ERR_RESOURCE_VSYNC)を返します。

(3) パラメータ詳細

vdc_output_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t                    tcon_half;
    uint16_t                    tcon_offset;
    const vdc_lcd_tcon_timing_t  * outctrl[VDC_LCD_TCONSIG_NUM];
    vdc_edge_t                  outcnt_lcd_edge;
    vdc_onoff_t                 out_endian_on;
    vdc_onoff_t                 out_swap_on;
    vdc_lcd_outformat_t          out_format;
    vdc_lcd_clkfreqsel_t         out_frq_sel;
    vdc_lcd_scan_t              out_dir_sel;
    vdc_lcd_clkphase_t           out_phase;
    uint32_t                    bg_color;
} vdc_output_t;
```

型 メンバ名	説明
uint16_t tcon_half	1/2fH タイミング設定 垂直カウンタのカウント動作タイミングを水平同期信号の立ち上がりからのクロック数を指定 0x0000 ~ 0x07FF
uint16_t tcon_offset	オフセット付き水平同期信号のタイミング設定 水平同期信号の立ち上がりからのクロック数を設定 0x0000 ~ 0x07FF
const vdc_lcd_tcon_timing_t * outctrl[VDC_LCD_TCONSIG_NUM]	LCD TCON のタイミング設定 使用しない信号に対しては NULL を指定してください。
vdc_edge_t outcnt_lcd_edge	LCD_DATA23 ~ 0 端子の出力位相制御 • VDC_EDGE_RISING: LCD_CLK 端子の立ち上がりエッジで出力 • VDC_EDGE_FALLING: LCD_CLK 端子の立ち下がりエッジで出力
vdc_onoff_t out_endian_on	ビットエンディアン変更 • VDC_OFF • VDC_ON
vdc_onoff_t out_swap_on	B/R 信号入れ替え • VDC_OFF • VDC_ON
vdc_lcd_outformat_t out_format	出力フォーマット選択 • VDC_LCD_OUTFORMAT_RGB888 (0): RGB888 • VDC_LCD_OUTFORMAT_RGB666 (1): RGB666 • VDC_LCD_OUTFORMAT_RGB565 (2): RGB565 • VDC_LCD_OUTFORMAT_SERIAL_RGB (3): シリアル RGB
vdc_lcd_clkfreqsel_t out_frq_sel	クロック周波数制御 • VDC_LCD_PARALLEL_CLKFRQ_1 (0): 1 倍速 (パラレル RGB) • VDC_LCD_SERIAL_CLKFRQ_3 (1): 3 倍速 (シリアル RGB) • VDC_LCD_SERIAL_CLKFRQ_4 (2): 4 倍速 (シリアル RGB)

	このパラメータは out_format に VDC_LCD_OUTFORMAT_SERIAL_RGB が指定されたの時のみ参照されます。この時、1 倍速の設定は禁止です。
vdc_lcd_scan_t out_dir_sel	スキャン方向選択 • VDC_LCD_SERIAL_SCAN_FORWARD (0): 正スキャン • VDC_LCD_SERIAL_SCAN_REVERSE (1): 逆スキャン このパラメータは out_format に VDC_LCD_OUTFORMAT_SERIAL_RGB が指定されたの時のみ参照されます。
vdc_lcd_clkphase_t out_phase	シリアル RGB 出力時のクロック位相調整 • VDC_LCD_SERIAL_CLKPHASE_0 (0): 0[クロック] • VDC_LCD_SERIAL_CLKPHASE_1 (1): 1[クロック] • VDC_LCD_SERIAL_CLKPHASE_2 (2): 2[クロック] • VDC_LCD_SERIAL_CLKPHASE_3 (3): 3[クロック] このパラメータは out_format に VDC_LCD_OUTFORMAT_SERIAL_RGB が指定されたの時のみ参照されます。また、out_frq_sel に VDC_LCD_SERIAL_CLKFRQ_3 が指定された場合、VDC_LCD_SERIAL_CLKPHASE_3 は設定禁止です。
uint32_t bg_color	背景色 RGB888 形式 (LSB 詰め)で指定してください。

vdc_lcd_tcon_sigsel_t は LCD パネル駆動用の各種タイミング信号 (LCD TCON)を表す列挙型です。

```
typedef enum
{
    VDC_LCD_TCONSIG_STVA_VS = 0,
    VDC_LCD_TCONSIG_STVB_VE,
    VDC_LCD_TCONSIG_STH_SP_HS,
    VDC_LCD_TCONSIG_STB_LP_HE,
    VDC_LCD_TCONSIG_CPV_GCK,
    VDC_LCD_TCONSIG_POLA,
    VDC_LCD_TCONSIG_POLB,
    VDC_LCD_TCONSIG_DE,
    VDC_LCD_TCONSIG_NUM
} vdc_lcd_tcon_sigsel_t;
```

列挙定数	値	説明
VDC_LCD_TCONSIG_STVA_VS	0	ゲートスタート信号、垂直同期信号 (STVA/VS)
VDC_LCD_TCONSIG_STVB_VE	1	ゲートスタート信号、垂直イネーブル信号 (STVB/VE)
VDC_LCD_TCONSIG_STH_SP_HS	2	ソーススタート信号、水平同期信号 (STH/SP/HS)
VDC_LCD_TCONSIG_STB_LP_HE	3	ソースストローブ信号、水平イネーブル信号 (STB/LP/HE)
VDC_LCD_TCONSIG_CPV_GCK	4	ゲートクロック信号 (CPV/GCK)
VDC_LCD_TCONSIG_POLA	5	VCOM 電圧極性制御信号 (POLA)
VDC_LCD_TCONSIG_POLB	6	VCOM 電圧極性制御信号 (POLB)
VDC_LCD_TCONSIG_DE	7	データイネーブル信号 (DE)
VDC_LCD_TCONSIG_NUM	8	LCD パネル駆動用信号タイプ数

vdc_lcd_tcon_timing_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t              tcon_hsvs;
    uint16_t              tcon_hwvw;
    vdc_lcd_tcon_polmode_t tcon_md;
    vdc_lcd_tcon_refsel_t  tcon_hs_sel;
    vdc_sig_pol_t         tcon_inv;
    vdc_lcd_tcon_pin_t     tcon_pin;
    vdc_edge_t            outcnt_edge;
} vdc_lcd_tcon_timing_t;
```

型 メンバ名	説明
uint16_t tcon_hsvs	信号のパルス開始位置 (第 1 の変化タイミング) 基準信号の立ち上がりから tcon_hsvs 後にパルス出力開始 0x0000 ~ 0x07FF [クロック周期、1/2fH 周期] POLA/POLB 信号を使用する時に tcon_md に VDC_LCD_TCON_POLMD_NORMAL 以外が指定された場合、'1'以上を設定してください。
uint16_t tcon_hwvw	信号のパルス幅 (第 2 の変化タイミング) tcon_hwvw 期間パルス出力 0x0000 ~ 0x07FF [クロック周期、1/2fH 周期]
vdc_lcd_tcon_polmode_t tcon_md	POLA/POLB 信号の生成モード選択 • VDC_LCD_TCON_POLMD_NORMAL (0): ノーマルモード 水平周期に 2 回変化する信号を生成します。 • VDC_LCD_TCON_POLMD_1X1REV (1): 1x1 リバースモード 1 水平周期ごとに極性が反転する信号を生成します。 • VDC_LCD_TCON_POLMD_1X2REV (2): 1x2 リバースモード 開始 1 水平期間で極性が反転し、その後、2 水平周期ごとに極性が反転する信号を生成します。 • VDC_LCD_TCON_POLMD_2X2REV (3): 2x2 リバースモード 2 水平周期ごとに極性が反転する信号を生成します。
vdc_lcd_tcon_refsel_t tcon_hs_sel	水平信号の動作基準選択 • VDC_LCD_TCON_REFSEL_HSYNC (0): 水平同期信号基準 • VDC_LCD_TCON_REFSEL_OFFSET_H (1): オフセット後の水平同期信号基準
vdc_sig_pol_t tcon_inv	信号の極性反転制御 • VDC_SIG_POL_NOT_INVERTED: 非反転 (正極性) • VDC_SIG_POL_INVERTED: 反転 (負極性)
vdc_lcd_tcon_pin_t tcon_pin	LCD TCON 出力端子選択

	• VDC_LCD_TCON_PIN_NON (-1): 出力なし • VDC_LCD_TCON_PIN_0 (0): LCD_TCON0 出力 • VDC_LCD_TCON_PIN_1 (1): LCD_TCON1 出力 • VDC_LCD_TCON_PIN_2 (2): LCD_TCON2 出力 • VDC_LCD_TCON_PIN_3 (3): LCD_TCON3 出力 • VDC_LCD_TCON_PIN_4 (4): LCD_TCON4 出力 • VDC_LCD_TCON_PIN_5 (5): LCD_TCON5 出力 • VDC_LCD_TCON_PIN_6 (6): LCD_TCON6 出力
vdc_edge_t outcnt_edge	信号の出力位相制御 • VDC_EDGE_RISING: LCD_CLK 端子の立ち上がりエッジで出力 • VDC_EDGE_FALLING: LCD_CLK 端子の立ち下がりエッジで出力

vdc_lcd_tcon_timing_t 構造体は、設定する LCD パネル駆動用の信号によって参照されるメンバが変わります。以下に各信号設定時のメンバの有効／無効を示します。無効なメンバは参照されません。

表 6-3　LCD パネル駆動信号の有効パラメータ

LCD パネル駆動用信号	vdc_lcd_tcon_timing_t 構造体のメンバ						
	tcon_hsvs	tcon_hwvw	tcon_md	tcon_hs_sel	tcon_inv	tcon_pin	outcnt_edge
STVA/VS	valid	valid	-	-	valid	valid	valid
STVB/VE	valid	valid	-	-	valid	valid	valid
STH/SP/HS	valid	valid	-	valid	valid	valid	valid
STB/LP/HE	valid	valid	-	valid	valid	valid	valid
CPV/GCK	valid	valid	-	valid	valid	valid	valid
POLA	valid	valid	valid	valid	valid	valid	valid
POLB	valid	valid	valid	valid	valid	valid	valid
DE	-	-	-	-	valid	valid	valid

【注】 valid: 設定値が参照される有効なメンバ。
-: 設定値が参照されないメンバ。

6.6 R_VDC_CallbackISR

概要	割り込みコールバック設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_CallbackISR(const vdc_channel_t ch, const vdc_int_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_int_t * param: 割り込みコールバック設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_RESOURCE_CLK: クロックリソースエラー — VDC_ERR_RESOURCE_VSYNC: 垂直同期信号リソースエラー

詳細

(1) 機能

本関数では、VDC の割り込みに関する以下の処理を行います。

- 割り込みコールバック関数のポインタが指定された場合、対応する割り込みのイネーブル設定
- 指定された割り込みコールバック関数の登録
- 割り込みコールバック関数のポインタが指定されなかった場合、対応する割り込みのディセーブル設定

本ドライバでは、関数「R_VDC_Initialize」と「R_VDC_Terminate」において、VDC の全ての割り込みをディセーブルとし、コールバック関数の登録も抹消されます。また、本関数「R_VDC_CallbackISR」を設定済みの割込みに対して適用することで、設定を上書くことも可能です。

(2) 使用条件

本関数の使用時にはパネルクロックと同期信号が設定されている必要があります。パネルクロックが設定されていない場合、クロックリソースエラー (VDC_ERR_RESOURCE_CLK)を返し、同期信号が設定されていない場合、垂直同期信号リソースエラー (VDC_ERR_RESOURCE_VSYNC)を返します。

(3) パラメータ詳細

vdc_int_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_int_type_t     type;
    void           (* callback)(vdc_int_type_t);
    uint16_t           line_num;
} vdc_int_t;
```

型 メンバ名	説明
vdc_int_type_t type	VDC 割り込みタイプ 詳細は「5.2(8)」を参照ください。
void (* callback)(vdc_int_type_t)	割り込みコールバック関数ポインタ type で指定した割り込みに対応する割り込みコールバック関数のポインタを指定します。コールバック関数はユーザが実装する必要があります。 コールバック関数が指定された割り込みは、割り込み処理が有効となります。callback に'0'が指定された場合、type で指定された割り込みは無効となります。
uint16_t line_num	ライン割り込み設定 画像のライン位置が line_num に一致した時に割り込み信号を出力します。 このパラメータは type に以下のライン割り込み指定された場合のみ有効です。 — VDC_INT_TYPE_VLINE — VDC_INT_TYPE_S0_WLINE

6.7　R_VDC_WriteDataControl

概要	データ書き込み制御処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_WriteDataControl(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_write_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_WR: レイヤ 0　書き込み処理 本ドライバでは必ずレイヤ 0 書込み処理を指定してください。 • vdc_write_t * param: データ書き込み制御パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_INPUT: 入力信号リソースエラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数では、データ書き込み制御に関する以下の処理を行います。

- 入力画像の取り込み領域の設定
- 入力画像の縮小制御／回転制御の設定
- フレームバッファ書き込み制御の設定

(2)　使用条件

本関数の使用前に関数「R_VDC_VideoInput」を呼び出すことで、ビデオ入力を有効にする必要があります。ビデオ入力が無効な場合、入力信号リソースエラー (VDC_ERR_RESOURCE_INPUT)を返します。

本関数の使用時に layer_id にて指定されたレイヤが既に有効な場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。関数「R_VDC_ReleaseDataControl」を呼び出すことで有効なレイヤを無効にすることができます。

垂直縮小処理は、垂直拡大処理と排他制御となっています。垂直縮小と垂直拡大の設定が同時に指定された場合の動作は保証しません。垂直拡大処理の設定は、関数「R_VDC_ReadDataControl」と「R_VDC_ChangeReadProcess」で行われます。

(3) パラメータ詳細

vdc_write_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_scalingdown_rot_t  scalingdown_rot;
    vdc_wr_rd_swa_t       res_wrswa;
    vdc_res_md_t          res_md;
    vdc_bst_md_t          res_bst_md;
    vdc_res_inter_t       res_inter;
    vdc_res_fs_rate_t     res_fs_rate;
    vdc_res_fld_sel_t     res_fld_sel;
    vdc_onoff_t           res_dth_on;
    void                * base;
    uint32_t              ln_off;
    uint32_t              flm_num;
    uint32_t              flm_off;
    void                * btm_base;
} vdc_write_t;
```

型 メンバ名	説明
vdc_scalingdown_rot_t scalingdown_rot	縮小／回転パラメータ
vdc_wr_rd_swa_t res_wrswa	8 ビット/16 ビット/32 ビットスワップ設定 • VDC_WR_RD_WRSWA_NON (0): スワップなし 1-2-3-4-5-6-7-8 • VDC_WR_RD_WRSWA_8BIT (1): 8-bit スワップ 2-1-4-3-6-5-8-7 • VDC_WR_RD_WRSWA_16BIT (2): 16-bit スワップ 3-4-1-2-7-8-5-6 • VDC_WR_RD_WRSWA_16_8BIT (3): 16-bit + 8-bit スワップ 4-3-2-1-8-7-6-5 • VDC_WR_RD_WRSWA_32BIT (4): 32-bit スワップ 5-6-7-8-1-2-3-4 • VDC_WR_RD_WRSWA_32_8BIT (5): 32-bit + 8-bit スワップ 6-5-8-7-2-1-4-3 • VDC_WR_RD_WRSWA_32_16BIT (6): 32-bit + 16-bit スワップ 7-8-5-6-3-4-1-2 • VDC_WR_RD_WRSWA_32_16_8BIT (7): 32-bit + 16-bit + 8-bit スワップ 8-7-6-5-4-3-2-1
vdc_res_md_t res_md	フレームバッファ書き込み映像フォーマット • VDC_RES_MD_YCBCR422 (0): YCbCr422 • VDC_RES_MD_RGB565 (1): RGB565 • VDC_RES_MD_RGB888 (2): RGB888 • VDC_RES_MD_YCBCR444 (3): YCbCr444
vdc_bst_md_t res_bst_md	フレームバッファ書き込み転送のバースト長

	• VDC_BST_MD_32BYTE (0): 32 バイト転送 (4 バースト) • VDC_BST_MD_128BYTE (1): 128 バイト転送 (16 バースト)
vdc_res_inter_t res_inter	フィールド動作モード設定 • VDC_RES_INTER_PROGRESSIVE (0): プログレッシブ • VDC_RES_INTER_INTERLACE (1): インタレース
vdc_res_fs_rate_t res_fs_rate	書き込み間隔 • VDC_RES_FS_RATE_PER1 (0): 入力信号に対して 1/1 • VDC_RES_FS_RATE_PER2 (1): 入力信号に対して 1/2 • VDC_RES_FS_RATE_PER4 (2): 入力信号に対して 1/4 • VDC_RES_FS_RATE_PER8 (3): 入力信号に対して 1/8
vdc_res_fld_sel_t res_fld_sel	書き込みフィールド選択 このパラメータは res_fs_rate に指定された値が VDC_RES_FS_RATE_PER1 以外の場合のみ有効です。 • VDC_RES_FLD_SEL_TOP (0): TOP フィールド • VDC_RES_FLD_SEL_BOTTOM (1): BOTTOM フィールド
vdc_onoff_t res_dth_on	ディザ補正 • VDC_OFF: 四捨五入 • VDC_ON: 2x2 パターンディザ
void * base	フレームバッファのベースアドレス NULL は設定しないでください。 res_bst_md に指定された値 • VDC_BST_MD_32BYTE の時 32 バイトアライメントのアドレスを指定してください。 • VDC_BST_MD_128BYTE の時 128 バイトアライメントのアドレスを指定してください。
uint32_t ln_off	フレームバッファのラインオフセットアドレス 0x0000 ~ 0x7FFF res_bst_md に指定された値 • VDC_BST_MD_32BYTE の時 32 の倍数で指定してください。 • VDC_BST_MD_128BYTE の時 128 の倍数で指定してください。
uint32_t flm_num	書き込みフレームバッファのフレーム数 0x0000 ~ 0x03FF flm_num + 1 がフレーム数となります。 回転処理時は 2 フレーム('1')以上に設定してください。
uint32_t flm_off	フレームバッファのフレームオフセットアドレス 0x00000000 ~ 0x007FFFFF このパラメータはフレーム数が 1 面(flm_num が'0')の時は 無効です。 res_bst_md に指定された値 • VDC_BST_MD_32BYTE の時 32 の倍数で指定してください。 • VDC_BST_MD_128BYTE の時 128 の倍数で指定してください。

void * btm_base	BOTTOM のフレームバッファのベースアドレス 不要な場合は NULL を指定してください。 res_bst_md に指定された値 • VDC_BST_MD_32BYTE の時 32 バイトアライメントのアドレスを指定してください。 • VDC_BST_MD_128BYTE の時 128 バイトアライメントのアドレスを指定してください。

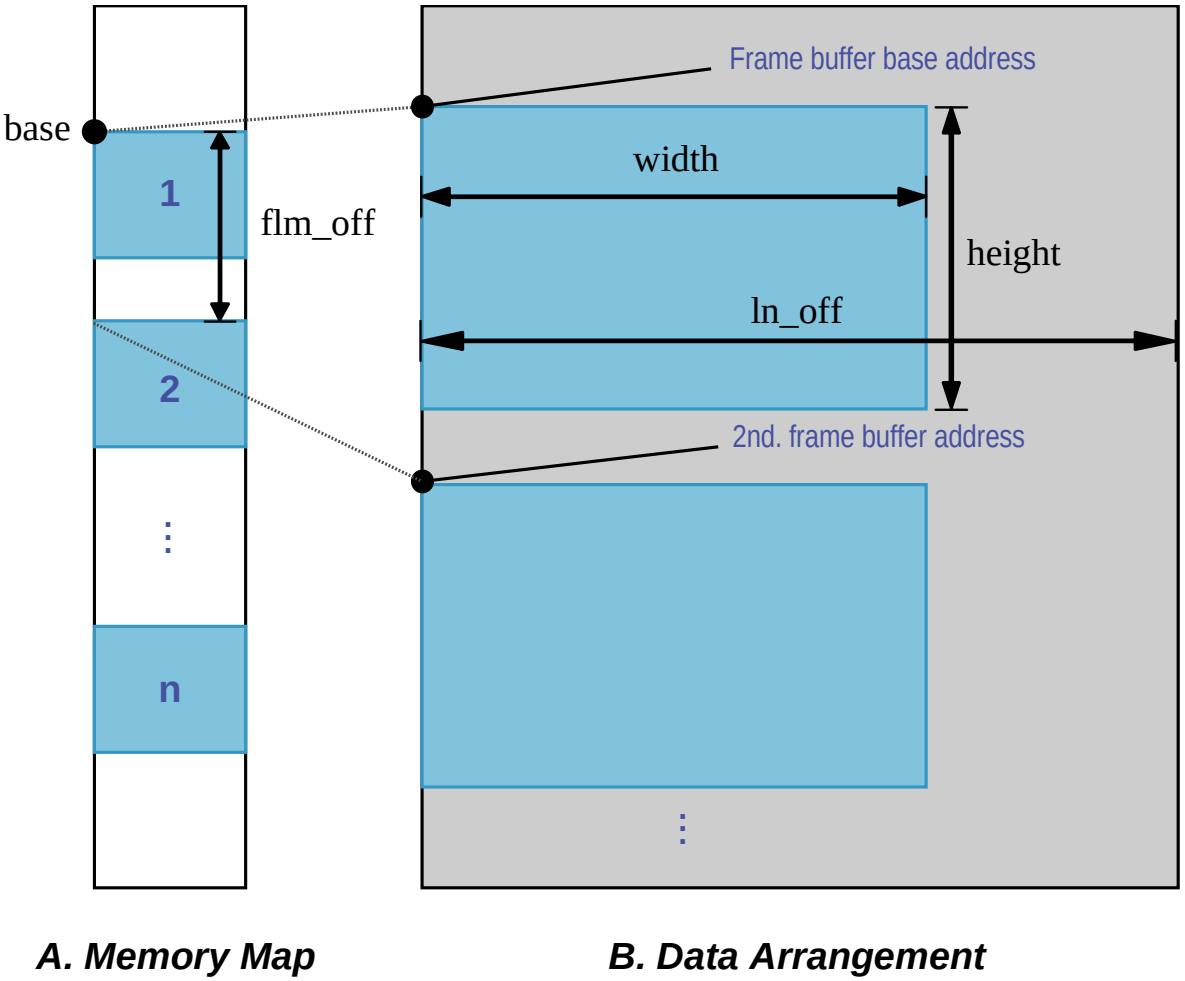
vdc_scalingdown_rot_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_period_rect_t  res;
    vdc_onoff_t       res_pfil_sel;
    uint16_t          res_out_vw;
    uint16_t          res_out_hw;
    vdc_onoff_t       adj_sel;
    vdc_wr_md_t       res_ds_wr_md;
} vdc_scalingdown_rot_t;
```

型 メンバ名	説明
vdc_period_rect_t res	画像取り込み範囲 vdc_period_rect_t 構造体については「5.3(1)」を参照ください。 res.vs は 4 ライン以上、(res.vs + res.vw)は 2039 ライン以内になるように設定してください。 res.hs は 16 クロック以上、(res.hs + res.hw)は 2015 クロック以内になるように設定してください。 取込映像信号垂直位置は res.vs + 1 になります。
vdc_onoff_t res_pfil_sel	輝度信号プリフィルタ • VDC_OFF • VDC_ON
uint16_t res_out_vw	縮小制御部出力の垂直有効ライン数 0x0000 ~ 0x07FF 4 ラインアライメント且つ res.vw 以下の大きさを指定してください。
uint16_t res_out_hw	縮小制御部出力の水平有効画素数 (映像クロック数) 0x0000 ~ 0x07FF 4 画素アライメント且つ res.hw 以下の大きさを指定してください。
vdc_onoff_t adj_sel	入力最終ライン欠落対策 縮小処理における入力最終ライン欠落の影響を低減する対策を実施するかを指定します。 • VDC_OFF • VDC_ON
vdc_wr_md_t res_ds_wr_md	フレームバッファ書き込み動作モード • VDC_WR_MD_NORMAL (0): 通常書き込み • VDC_WR_MD_MIRROR (1): 水平鏡像書き込み • VDC_WR_MD_ROT_90DEG (2): 90 度回転書き込み • VDC_WR_MD_ROT_180DEG (3): 180 度回転書き込み

- VDC_WR_MD_ROT_270DEG (4): 270 度回転書き込み
90 度、180 度、270 度回転書き込みは、フレームバッファ書き込み映像フォーマット(res_md)が YCbCr422 か RGB565 の時のみ有効です。

フレームバッファに関するパラメータの指定とメモリ配置の関係を図 6-2 に示します。



【注】 base: フレームバッファのベースアドレス
flm_off: フレームバッファのフレームオフセットアドレス [byte]
n: フレームバッファ数 (= flm_num + 1)
height: 垂直有効ライン数 [line]
width: 水平有効画素数 [pixel]
ln_off: フレームバッファのラインオフセットアドレス [byte]

図 6-2　フレームバッファ関連パラメータとメモリ配置

フレームバッファに格納される画像の幅と高さ (図 6-2 の width と height)は、それぞれ縮小制御部出力の水平有効画素数 (res_out_hw)と垂直有効ライン数 (res_out_vw)に一致します。ただし、フレームバッファ書き込み動作モード (res_ds_wr_md)が 90 度、若しくは 270 度回転書込みの場合は、幅と高さはそれぞれ res_out_vw と res_out_hw となります。

6.8 R_VDC_ChangeWriteProcess

概要	データ書き込み変更処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ChangeWriteProcess(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_write_chg_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_WR: レイヤ 0　書き込み処理 本ドライバでは必ずレイヤ 0 書込み処理を指定してください。 • vdc_write_chg_t * param: データ書き込み変更パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1) 機能

本関数では、データ書き込み処理の動作中にデータ書き込み制御に関する以下の処理を行います。

- 入力画像の取り込み領域の変更
- 入力画像の縮小制御／回転制御の変更

(2) 使用条件

本関数の使用時に layer_id にて指定されたレイヤが以下の条件に該当しない場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

- 指定されたレイヤが有効
- 指定されたレイヤが動作中

関数「R_VDC_WriteDataControl」を呼び出すことで、レイヤが有効になります。また、関数「R_VDC_StartProcess」を呼び出すことで、停止中のレイヤを動作させることができます。

垂直縮小処理は、垂直拡大処理と排他制御となっています。垂直縮小と垂直拡大の設定が同時に指定された場合の動作は保証しません。垂直拡大処理の設定は、関数「R_VDC_ReadDataControl」と「R_VDC_ChangeReadProcess」で行われます。

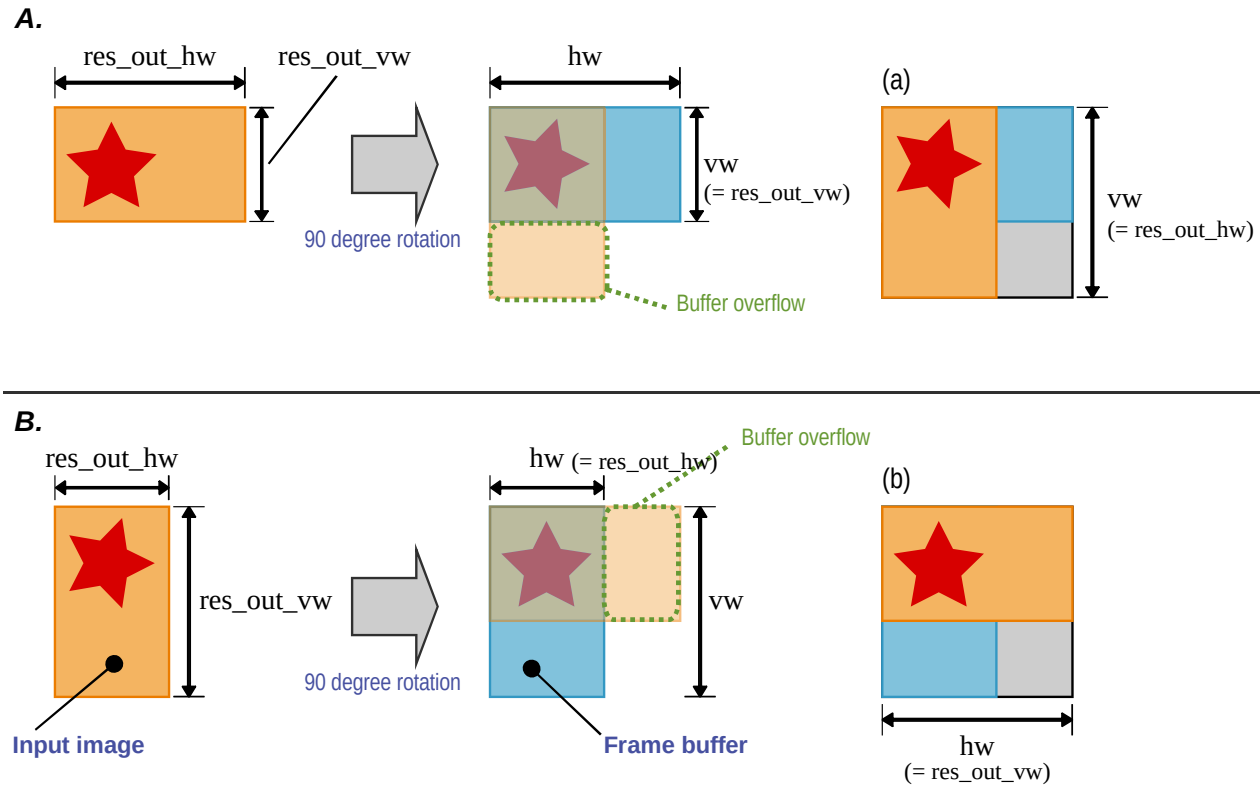
(3)　パラメータ詳細

vdc_write_chg_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_scalingdown_rot_t   scalingdown_rot;
} vdc_write_chg_t;
```

型 メンバ名	説明
vdc_scalingdown_rot_t scalingdown_rot	縮小／回転パラメータ 詳細は「6.7」のパラメータ詳細を参照ください。

本関数を使用すると、データ書き込み処理の動作中に回転制御を変更することができます。変更前に対して回転角度を 90 度、若しくは 270 度変更する場合、データ書き込みに必要なフレームバッファのサイズが変わる可能性があります。



【注】　res_out_vw: 縮小後の画像垂直幅　　res_out_hw: 縮小後の画像水平幅
　　vw: フレームバッファの垂直幅　　hw: フレームバッファの水平幅

図 6-3　回転処理とフレームバッファサイズ

図 6-3 は入力画像を 90 度回転した時に生じるフレームバッファサイズの変化について 2 つのケースを表したものです。回転前の時点では、データ書き込みに必要なフレームバッファの幅 (hw)と高さ (vw)は、そ

れぞれ縮小後の幅 (res_out_hw)と高さ (res_out_vw)です。90 度の回転によりこれらのサイズが変化する為、予期しない領域へデータが書き込まれる可能性があります。これを回避する為には、予めフレームバッファを大きく確保しておく必要があります(図中、(a)及び(b))。

6.9 R_VDC_ReadDataControl

概要	データ読み出し制御処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ReadDataControl(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_read_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_read_t * param: データ読み出し制御パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_PARAM_CONDITION: 不許可条件エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1) **機能**

本関数では、データ読み出し制御に関する以下の処理を行います。

- グラフィックス画像の表示領域領域の設定
- 画像の拡大制御の設定 (レイヤ 0 のみ)
- フレームバッファ読み出し制御の設定

(2) **使用条件**

本関数の使用時に layer_id にて指定されたレイヤが既に有効な場合、本関数はレイヤリソースエラー(VDC_ERR_RESOURCE_LAYER)を返します。関数「R_VDC_ReleaseDataControl」を呼び出すことで有効なレイヤを無効にすることができます。

垂直縮小処理と排他制御となっています。垂直縮小と垂直拡大の設定が同時に指定された場合の動作は保証しません。垂直縮小処理の設定は、関数「R_VDC_WriteDataControl」と「R_VDC_ChangeWriteProcess」で行われます。

(3) パラメータ詳細

vdc_read_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_gr_ln_off_dir_t          gr_ln_off_dir;
    vdc_gr_flm_sel_t             gr_flm_sel;
    vdc_onoff_t                  gr_imr_flm_inv;
    vdc_bst_md_t                 gr_bst_md;
    void                       * gr_base;
    uint32_t                     gr_ln_off;
    const vdc_width_read_fb_t    * width_read_fb;
    vdc_onoff_t                  adj_sel;
    vdc_gr_format_t              gr_format;
    vdc_gr_ycc_swap_t            gr_ycc_swap;
    vdc_wr_rd_swa_t              gr_rdswa;
    vdc_period_rect_t            gr_grc;
} vdc_read_t;
```

型 メンバ名	説明
vdc_gr_ln_off_dir_t gr_ln_off_dir	フレームバッファのラインオフセットアドレスの方向設定 • VDC_GR_LN_OFF_DIR_INC (0): ラインオフセットアドレス分をインクリメント • VDC_GR_LN_OFF_DIR_DEC (1): ラインオフセットアドレス分をデクリメント
vdc_gr_flm_sel_t gr_flm_sel	フレームバッファアドレス設定信号の選択 • VDC_GR_FLM_SEL_SCALE_DOWN (0): 縮小処理と連携 • VDC_GR_FLM_SEL_FLM_NUM (1): フレーム 0 を選択 • VDC_GR_FLM_SEL_POINTER_BUFF (3): ポインタバッファと連携
vdc_onoff_t gr_imr_flm_inv	歪み補正フレームバッファ番号設定 このパラメータは本ドライバでは参照されません。 VDC_OFF を設定してください。
vdc_bst_md_t gr_bst_md	フレームバッファバースト転送モード • VDC_BST_MD_32BYTE (0): 32 バイト転送 • VDC_BST_MD_128BYTE (1): 128 バイト転送
void * gr_base	フレームバッファのベースアドレス gr_flm_sel に指定された値が VDC_GR_FLM_SEL_POINTER_BUFF 以外の時、NULL は設定しないでください。
uint32_t gr_ln_off	フレームバッファのラインオフセットアドレス 0x0000 ~ 0x7FFF gr_bst_md に指定された値 • VDC_BST_MD_32BYTE の時 32 の倍数で指定してください。

	• VDC_BST_MD_128BYTE の時 128 の倍数で指定してください。
const vdc_width_read_fb_t * width_read_fb	読み出しフレームバッファサイズ NULL を指定した場合、グラフィックス表示領域の幅(gr_grc.hw)と高さ(gr_grc.vw)と同じサイズであると看做されます。
vdc_onoff_t adj_sel	折り返し対策 拡大処理における折り返し画素の影響を低減する対策を実施するかを指定します。 • VDC_OFF • VDC_ON
vdc_gr_format_t gr_format	フレームバッファ読み出し信号のフォーマット • VDC_GR_FORMAT_RGB565 (0): RGB565 • VDC_GR_FORMAT_RGB888 (1): RGB888 • VDC_GR_FORMAT_ARGB1555 (2): ARGB1555 • VDC_GR_FORMAT_ARGB4444 (3): ARGB4444 • VDC_GR_FORMAT_ARGB8888 (4): ARGB8888 • VDC_GR_FORMAT_CLUT8 (5): CLUT8 • VDC_GR_FORMAT_CLUT4 (6): CLUT4 • VDC_GR_FORMAT_CLUT1 (7): CLUT1 • VDC_GR_FORMAT_YCBCR422 (8): YCbCr422 ※ • VDC_GR_FORMAT_YCBCR444 (9): YCbCr444 ※ • VDC_GR_FORMAT_RGBA5551 (10): RGBA5551 • VDC_GR_FORMAT_RGBA8888 (11): RGBA8888
vdc_gr_ycc_swap_t gr_ycc_swap	YCbCr422 フォーマット時バッファ読み出しデータのスワップ制御 このパラメータは gr_format に指定された値が VDC_GR_FORMAT_YCBCR422 の場合のみ有効です。 • VDC_GR_YCCSWAP_CBY0CRY1 (0): CbY0/Cr/Y1 • VDC_GR_YCCSWAP_Y0CBY1CR (1): Y0/Cb/Y1/Cr • VDC_GR_YCCSWAP_CRY0CBY1 (2): Cr/Y0/Cb/Y1 • VDC_GR_YCCSWAP_Y0CRY1CB (3): Y0/Cr/Y1/Cb • VDC_GR_YCCSWAP_Y1CRY0CB (4): Y1/Cr/Y0/Cb • VDC_GR_YCCSWAP_CRY1CBY0 (5): Cr/Y1/Cb/Y0 • VDC_GR_YCCSWAP_Y1CBY0CR (6): Y1/Cb/Y0/Cr • VDC_GR_YCCSWAP_CBY1CRY0 (7): Cb/Y1/Cr/Y0
vdc_wr_rd_swa_t gr_rdswa	8 ビット/16 ビット/32 ビットスワップ設定 • VDC_WR_RD_WRSWA_NON (0): スワップなし 1-2-3-4-5-6-7-8 • VDC_WR_RD_WRSWA_8BIT (1): 8-bit スワップ 2-1-4-3-6-5-8-7 • VDC_WR_RD_WRSWA_16BIT (2): 16-bit スワップ 3-4-1-2-7-8-5-6 • VDC_WR_RD_WRSWA_16_8BIT (3): 16-bit + 8-bit スワップ 4-3-2-1-8-7-6-5 • VDC_WR_RD_WRSWA_32BIT (4): 32-bit スワップ 5-6-7-8-1-2-3-4 • VDC_WR_RD_WRSWA_32_8BIT (5): 32-bit + 8-bit スワップ 6-5-8-7-2-1-4-3 • VDC_WR_RD_WRSWA_32_16BIT (6): 32-bit + 16-bit スワップ 7-8-5-6-3-4-1-2

	• VDC_WR_RD_WRSWA_32_16_8BIT (7): 32-bit + 16-bit + 8-bit スワップ 8-7-6-5-4-3-2-1
vdc_period_rect_t gr_grc	グラフィックス表示領域 vdc_period_rect_t 構造体については「5.3(1)」を参照ください。 gr_grc.vs は 4 ライン以上、(gr_grc.vs + gr_grc.vw)は 2039 ライン以内になるように設定してください。 gr_grc.hs は 16 クロック以上、(gr_grc.hs + gr_grc.hw)は 2015 クロック以内になるように設定してください。

【注】 YCbCr422 と YCbCr444 はグラフィックス 0 のみ指定可能です。

フレームバッファアドレス設定信号の選択 (gr_flm_sel)は、レイヤ毎に指定可能なパラメータが異なります。各レイヤの設定可能な値については表 6-4 を参照ください。

表 6-4　設定可能なフレームバッファアドレス設定信号の選択

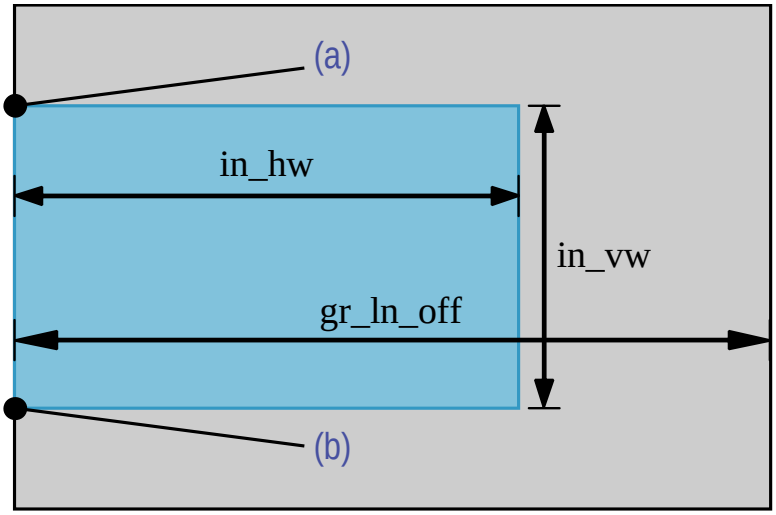
レイヤ ID	設定可能な値
VDC_LAYER_ID_0_RD	VDC_GR_FLM_SEL_SCALE_DOWN VDC_GR_FLM_SEL_FLM_NUM VDC_GR_FLM_SEL_POINTER_BUFF
VDC_LAYER_ID_2_RD	VDC_GR_FLM_SEL_FLM_NUM
VDC_LAYER_ID_3_RD	VDC_GR_FLM_SEL_FLM_NUM

vdc_width_read_fb_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t              in_vw;
    uint16_t              in_hw;
} vdc_width_read_fb_t;
```

型 メンバ名	説明
uint16_t in_vw	1 フレームのライン数設定 0x0000 ~ 0x07FF
uint16_t in_hw	水平有効期間の幅設定 0x0000 ~ 0x07FF

データ読み出し処理のフレームバッファに関するパラメータの指定とメモリ配置の関係を図 6-4 に示します。



【注】 (a), (b): フレームバッファのベースアドレス (gr_base)
in_vw: 1 フレームのライン数 [line]
in_hw: 水平有効期間の幅 [pixel]
gr_ln_off: フレームバッファのラインオフセットアドレス [byte]

図 6-4　読み出し処理のフレームバッファ関連パラメータとメモリ配置

フレームバッファのベースアドレス (gr_base)は、フレームバッファのラインオフセットアドレスの方向設定 (gr_ln_off_dir)に従って設定を変更する必要があります。

- gr_ln_off_dir が VDC_GR_LN_OFF_DIR_INC の時
  フレームバッファの先頭から順にデータが読み出されます。gr_base へ、フレームバッファの先頭アドレス (図 6-4、(a))を指定してください。
- gr_ln_off_dir が VDC_GR_LN_OFF_DIR_DEC の時
  フレームバッファの最終行からデータが読み出されます。gr_base へ、フレームバッファの最終ラインのアドレス (図 6-4、(b))を指定してください。

6.10　R_VDC_ChangeReadProcess

概要	データ読み出し変更処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ChangeReadProcess(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_read_chg_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_read_chg_t * param: データ読み出し変更パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　**機能**

本関数では、データ読み出し処理の動作中にデータ読み出し制御に関する以下の処理を行います。

- フレームバッファアドレスの変更
- フレームバッファ読み出しサイズ (画像の拡大制御)の変更 (レイヤ 0 のみ)
- グラフィックス画像の表示領域領域の設定
- グラフィックス表示設定の変更

(2)　使用条件

本関数の使用時に layer_id にて指定されたレイヤが以下の条件に該当しない場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

- 指定されたレイヤが有効
- 指定されたレイヤが動作中

関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。また、関数「R_VDC_StartProcess」を呼び出すことで、停止中のレイヤを動作させることができます。

垂直拡大処理は、垂直縮小処理と排他制御となっています。垂直縮小と垂直拡大の設定が同時に指定された場合の動作は保証しません。垂直縮小処理の設定は、関数「R_VDC_WriteDataControl」と「R_VDC_ChangeWriteProcess」で行われます。

(3)　パラメータ詳細

vdc_read_chg_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    void                        * gr_base;
    const vdc_width_read_fb_t    * width_read_fb;
    const vdc_period_rect_t      * gr_grc;
    const vdc_gr_disp_sel_t      * gr_disp_sel;
} vdc_read_chg_t;
```

型 メンバ名	説明
void * gr_base	フレームバッファのベースアドレス 変更しない場合は NULL を設定してください。
const vdc_width_read_fb_t * width_read_fb	読み出しフレームバッファサイズ vdc_width_read_fb_t 構造体については「6.9」を参照ください。 変更しない場合は NULL を設定してください。
const vdc_period_rect_t * gr_grc	グラフィックス表示領域 vdc_period_rect_t 構造体については「5.3(1)」を参照ください。 変更しない場合は NULL を設定してください。
const vdc_gr_disp_sel_t * gr_disp_sel	グラフィックス表示設定 詳細は「5.2(9)」、及び「6.11(3)」を参照ください。 変更しない場合は NULL を設定してください。

vdc_read_chg_t 構造体の各パラメータについて、NULL が指定された場合は設定は変更されません。gr_base、width_read_fb、gr_grc の各パラメータは関数「R_VDC_ReadDataControl」にて設定された値、gr_disp_sel のパラメータは関数「R_VDC_StartProcess」にて以前に設定された値のままとなります。

6.11　R_VDC_StartProcess

概要	データ書き込み／読み出し開始処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_StartProcess(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_start_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_ALL: 有効なレイヤ全て — VDC_LAYER_ID_0_WR: レイヤ 0　書き込み処理 — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_start_t * param: データ書き込み／読み出し開始パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　**機能**

本関数は、レイヤの動作開始処理を行います。layer_id にて指定されたレイヤが VDC_LAYER_ID_ALL の場合、停止中且つ有効な全てのレイヤの動作を開始します。VDC_LAYER_ID_ALL 以外の場合は指定されたレイヤのみ動作を開始します。

書き込み処理に関する開始では、フレームバッファに対する書き込み動作を開始します。読み出し処理に関する開始では、フレームバッファからの読み出し動作を開始し、各レイヤのグラフィックス表示設定を指定された値に設定します。

(2) 使用条件

本関数の使用時に layer_id にて VDC_LAYER_ID_ALL が指定された場合、使用条件はありません。それ以外の場合は以下のような使用条件があります。これらの条件に該当しない場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

- 指定されたレイヤが有効
- 指定されたレイヤが停止中

書き込み処理に関するレイヤには関数「R_VDC_WriteDataControl」、読み出し処理に関するレイヤには関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。また、関数「R_VDC_StopProcess」を呼び出すことで、動作中のレイヤを停止させることができます。

(3) パラメータ詳細

vdc_start_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    const vdc_gr_disp_sel_t     * gr_disp_sel;
} vdc_start_t;
```

型 メンバ名	説明
const vdc_gr_disp_sel_t * gr_disp_sel	グラフィックス表示設定 詳細は「5.2(9)」、及び以下の説明を参照ください。 変更しない場合は NULL を設定してください。 本設定はグラフィックス部 (読み出し処理)の設定です。 layer_id に書き込み処理のレイヤが指定された場合、本設定は無効です。

gr_disp_sel の設定については以下の設定例を参照ください。

1. layer_id に VDC_LAYER_ID_ALL を指定する場合
   全てのグラフィックス部 (読み出し処理のレイヤ)に対するグラフィックス表示設定を指定します。
   変更が不要なレイヤに対しては、VDC_DISPSEL_IGNORED を指定します。

```
1       vdc_error_t           error;
2       vdc_gr_disp_sel_t      gr_disp_sel[VDC_GR_TYPE_NUM];
3       vdc_start_t           start;
4
5       gr_disp_sel[VDC_GR_TYPE_GR0] = VDC_DISPSEL_IGNORED;
6       gr_disp_sel[VDC_GR_TYPE_GR2] = VDC_DISPSEL_CURRENT;
7       gr_disp_sel[VDC_GR_TYPE_GR3] = VDC_DISPSEL_BLEND;
8
9       start.gr_disp_sel           = gr_disp_sel;
10
11      error = R_VDC_StartProcess(VDC_CHANNEL_0, VDC_LAYER_ID_ALL, &start);
```

2. layer_id に単一のレイヤを指定する場合
   指定されたグラフィックス部 (読み出し処理のレイヤ)に対するグラフィックス表示設定のみを指定します。下記の例はグラフィックス部 2 (レイヤ 2 の読み出しプロセス)に対する設定です。

```
1       vdc_error_t           error;
2       vdc_gr_disp_sel_t      gr_disp_sel;
3       vdc_start_t           start;
4
5       gr_disp_sel        = VDC_DISPSEL_CURRENT;
6
7       start.gr_disp_sel   = &gr_disp_sel;
8
9       error = R_VDC_StartProcess(VDC_CHANNEL_0, VDC_LAYER_ID_2_RD, &start);
```

各レイヤのグラフィックス表示設定は、関数「R_VDC_DisplayOutput」、「R_VDC_ReadDataControl」、及び「R_VDC_StopProcess」の呼び出し時にドライバにより初期化されます。本関数にて変更しない場合は、ドライバによる初期設定の値のままとなります。表 6-5 に各レイヤのグラフィックス表示設定の初期設定値を示します。

表 6-5　グラフィックス表示設定の初期設定値

レイヤ ID	初期設定値	説明
VDC_LAYER_ID_0_RD	VDC_DISPSEL_BACK	使用しない時は背景色表示
VDC_LAYER_ID_2_RD	VDC_DISPSEL_LOWER	使用しない時は下層表示
VDC_LAYER_ID_3_RD	VDC_DISPSEL_LOWER	使用しない時は下層表示

グラフィックス表示設定の VDC_DISPSEL_BACK は背景色表示、VDC_DISPSEL_CURRENT はグラフィックス表示の時に設定します。その他のグラフィックス表示設定は、使用レイヤとその用途に従い指定します。

表 6-6　グラフィックス表示設定と用途

レイヤ ID	設定値	用途
VDC_LAYER_ID_0_RD	VDC_DISPSEL_LOWER	• ビデオ入力映像表示 • グラフィックス拡大表示
	VDC_DISPSEL_BLEND	• クロマキー処理画像表示
VDC_LAYER_ID_2_RD / VDC_LAYER_ID_3_RD	VDC_DISPSEL_LOWER	• 下層グラフィックス表示
	VDC_DISPSEL_BLEND	• 下層グラフィックスとカレントグラフィックスの合成画像表示

6.12　R_VDC_StopProcess

概要	データ書き込み／読み出し停止処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_StopProcess(const vdc_channel_t ch, const vdc_layer_id_t layer_id);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_ALL: 有効なレイヤ全て — VDC_LAYER_ID_0_WR: レイヤ 0　書き込み処理 — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数は、動作中のレイヤの停止処理を行います。layer_id にて指定されたレイヤが VDC_LAYER_ID_ALL の場合、動作中且つ有効な全てのレイヤが停止します。VDC_LAYER_ID_ALL 以外の場合は指定されたレイヤのみが停止します。

書き込み処理に関する停止では、フレームバッファに対する書き込みを停止します。読み出し処理に関する停止では、フレームバッファからの読み出しを停止し、各レイヤのグラフィックス表示設定を初期状態に戻します。グラフィックス表示設定の初期状態については、表 6-5 を参照ください。

(2)　使用条件

本関数の使用時に layer_id にて VDC_LAYER_ID_ALL が指定された場合、使用条件はありません。それ以外の場合は以下のような使用条件があります。これらの条件に該当しない場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

- 指定されたレイヤが有効
- 指定されたレイヤが動作中

書き込み処理に関するレイヤには関数「R_VDC_WriteDataControl」、読み出し処理に関するレイヤには関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。また、関数「R_VDC_StartProcess」を呼び出すことで、停止中のレイヤを動作させることができます。

6.13　R_VDC_ReleaseDataControl

概要	データ書き込み／読み出し制御解放処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ReleaseDataControl(const vdc_channel_t ch, const vdc_layer_id_t layer_id);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_ALL: 有効なレイヤ全て — VDC_LAYER_ID_0_WR: レイヤ 0　書き込み処理 — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数では、以下の処理を行います。

- 指定されたレイヤの無効化

layer_id にて指定されたレイヤが VDC_LAYER_ID_ALL の場合、動作中で無い且つ有効な全てのレイヤが無効化されます。VDC_LAYER_ID_ALL 以外の場合は指定されたレイヤのみが無効化されます。

(2)　使用条件

本関数の使用時に layer_id にて VDC_LAYER_ID_ALL が指定された場合、使用条件はありません。それ以外の場合は以下のような使用条件があります。これらの条件に該当しない場合、本関数はレイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

- 指定されたレイヤが有効
- 指定されたレイヤが動作中ではない

書き込み処理に関するレイヤ ID には関数「R_VDC_WriteDataControl」、読み出し処理に関するレイヤに対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。また、関数「R_VDC_StopProcess」を呼び出すことで、動作中のレイヤを停止することができます。

6.14　R_VDC_VideoNoiseReduction

概要	ノイズリダクション設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_VideoNoiseReduction(const vdc_channel_t ch, const vdc_onoff_t nr1d_on, const vdc_noise_reduction_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_onoff_t nr1d_on: ノイズリダクション ON/OFF 設定 • vdc_noise_reduction_t * param: ノイズリダクション設定パラメータ NULL が指定された場合、設定は変更されません。ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については構造体の説明を参照ください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_RESOURCE_INPUT: 入力信号リソースエラー

詳細

(1)　機能

本関数では、ノイズリダクションに関する以下の処理を行います。

- ノイズリダクションの ON/OFF 設定
- Y/G 信号、Cb/B 信号、Cr/R 信号のノイズリダクションパラメータ設定

ノイズリダクションのパラメータ設定と ON/OFF の制御は別個に設定できます。一度設定されたノイズリダクションのパラメータの設定は、ハードウェアのリセットか、別の設定で上書きされるまでは有効です。

(2)　使用条件

本関数の使用前に、関数「R_VDC_VideoInput」を呼び出すことで、ビデオ入力を有効にする必要があります。ビデオ入力が無効な場合、入力信号リソースエラー (VDC_ERR_RESOURCE_INPUT)を返します。

(3) パラメータ詳細

vdc_noise_reduction_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_nr_param_t    y;
    vdc_nr_param_t    cb;
    vdc_nr_param_t    cr;
} vdc_noise_reduction_t;
```

型 メンバ名	説明
vdc_nr_param_t y	Y/G 信号のノイズリダクションパラメータ
vdc_nr_param_t cb	Cb/B 信号のノイズリダクションパラメータ
vdc_nr_param_t cr	Cr/R 信号のノイズリダクションパラメータ

vdc_nr_param_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_nr_tap_t   nr1d_tap;
    uint32_t       nr1d_th;
    vdc_nr_gain_t  nr1d_gain;
} vdc_nr_param_t;
```

型 メンバ名	初期値	説明
vdc_nr_tap_t nr1d_tap	0	TAP 選択 • VDC_NR_TAPSEL_1 (0): 1 画素隣接 • VDC_NR_TAPSEL_2 (1): 2 画素隣接 • VDC_NR_TAPSEL_3 (2): 3 画素隣接 • VDC_NR_TAPSEL_4 (3): 4 画素隣接
uint32_t nr1d_th	8	コアリングの最大値 (絶対値) 0x0000 ~ 0x007F
vdc_nr_gain_t nr1d_gain	3	ノイズリダクションゲイン調整 • VDC_NR_GAIN_1_2 (0): 1/2 • VDC_NR_GAIN_1_4 (1): 1/4 • VDC_NR_GAIN_1_8 (2): 1/8 • VDC_NR_GAIN_1_16 (3): 1/16

6.15 R_VDC_ImageColorMatrix

概要	カラーマトリクス設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ImageColorMatrix(const vdc_channel_t ch, const vdc_color_matrix_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_color_matrix_t * param: カラーマトリクス設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_CONDITION: 不許可条件エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

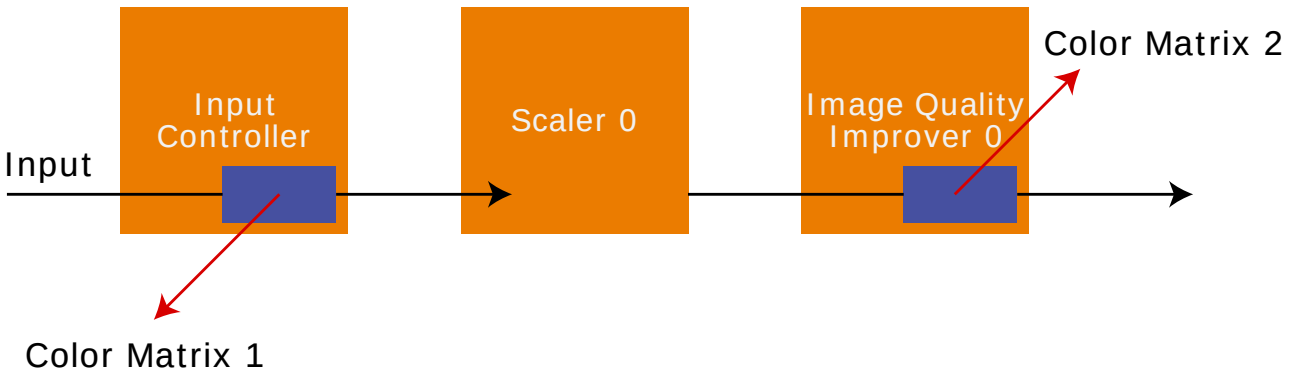
(1) 機能

本関数では、指定されたカラーマトリクスの設定を行います。RZ/A2M に搭載されている VDC にはカラーマトリクスが 2 箇所にあります(図 6-5 参照)。

カラーマトリクスは使用するカラーフォーマットに従い、VDC ドライバにより自動的に設定されます。従って、カラーマトリクスの値を動的に変更する必要がある場合を除いて、本関数を使用する必要はありません。VDC ドライバによるカラーマトリクスの自動的な設定は以下のように行われます。

- 関数「R_VDC_WriteDataControl」が呼び出された時。
  — ビデオ映像の入力フォーマットと関数「R_VDC_WriteDataControl」にて指定されたフレームバッファ書き込み映像フォーマットから必要な色変換を判定し、入力制御部のカラーマトリクスへ設定します。
- 関数「R_VDC_ReadDataControl」がレイヤ 0 の読み出しプロセスに対して呼び出された時。
  — 関数「R_VDC_ReadDataControl」にて指定されたフレームバッファ読み出し信号のフォーマットから必要な色変換を判定し、画質改善部 0 のカラーマトリクスへ設定します。

VDC ドライバにより自動的に設定されるカラーマトリクスの値については「5.5」を参照ください。



【注】 Color Matrix 1: 入力制御部のカラーマトリクス
Color Matrix 2: 画質改善部 0 のカラーマトリクス

図 6-5　カラーマトリクス

(2)　使用条件

入力制御部のカラーマトリクスへ設定する為に本関数を使用する場合、レイヤ 0 の書き込みプロセスが有効である必要があります。関数「R_VDC_WriteDataControl」を呼び出すことで、レイヤ 0 の書き込みプロセスが有効になります。

画質改善部 0 のカラーマトリクスへ設定する為に本関数を使用する場合、レイヤ 0 の読み出しプロセスが有効である必要があります。レイヤ 0 に対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤ 0 の読み出しプロセスが有効になります。

カラーマトリクスに対応するレイヤが無効な状態で本関数が呼び出された場合、レイヤリソースエラー(VDC_ERR_RESOURCE_LAYER)を返します。

(3) パラメータ詳細

vdc_color_matrix_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_colormtx_module_t  module;
    vdc_colormtx_mode_t    mtx_mode;
    uint16_t              offset[VDC_COLORMTX_OFFST_NUM];
    uint16_t              gain[VDC_COLORMTX_GAIN_NUM];
} vdc_color_matrix_t;
```

型 メンバ名	説明
vdc_colormtx_module_t module	カラーマトリクス設定対象モジュール選択 • VDC_COLORMTX_IMGCNT (0): 入力制御部 • VDC_COLORMTX_ADJ_0 (1): 画質改善部 0
vdc_colormtx_mode_t mtx_mode	カラーマトリクス動作モード • VDC_COLORMTX_GBR_GBR: GBR ⇒ GBR • VDC_COLORMTX_GBR_YCBCR: GBR ⇒ YCbCr ※ • VDC_COLORMTX_YCBCR_GBR: YCbCr ⇒ GBR • VDC_COLORMTX_YCBCR_YCBCR: YCbCr ⇒ YCbCr ※
uint16_t offset[VDC_COLORMTX_OFFST_NUM]	Y/G、B、R 信号のオフセット(DC)調整 0x0000 (-128) ~ 0x0080 (0) ~ 0x00FF (+127)
uint16_t gain[VDC_COLORMTX_GAIN_NUM]	GG、GB、GR、BG、BB、BR、RG、RB、RR のゲイン調整 符号付 (2 の補数) -1024 ~ +1023[LSB], 256[LSB] = 1.0[倍]

【注】 YCbCr へ変換する動作モードは、module に入力制御部 (VDC_COLORMTX_IMGCNT)を指定した場合のみ使用可能です。

vdc_colormtx_offset_t はカラーマトリクスのオフセットを表す列挙型です。

```
typedef enum
{
    VDC_COLORMTX_OFFST_YG = 0,
    VDC_COLORMTX_OFFST_B,
    VDC_COLORMTX_OFFST_R,
    VDC_COLORMTX_OFFST_NUM
} vdc_colormtx_offset_t;
```

列挙定数	値	説明
VDC_COLORMTX_OFFST_YG	0	Y/G 信号のオフセット (DC)
VDC_COLORMTX_OFFST_B	1	B 信号のオフセット (DC)
VDC_COLORMTX_OFFST_R	2	R 信号のオフセット (DC)
VDC_COLORMTX_OFFST_NUM	3	カラーマトリクスのオフセットパラメータ数

vdc_colormtx_gain_t はカラーマトリクスのゲインを表す列挙型です。

```
typedef enum
{
    VDC_COLORMTX_GAIN_GG = 0,
    VDC_COLORMTX_GAIN_GB,
    VDC_COLORMTX_GAIN_GR,
    VDC_COLORMTX_GAIN_BG,
    VDC_COLORMTX_GAIN_BB,
    VDC_COLORMTX_GAIN_BR,
    VDC_COLORMTX_GAIN_RG,
    VDC_COLORMTX_GAIN_RB,
    VDC_COLORMTX_GAIN_RR,
    VDC_COLORMTX_GAIN_NUM
} vdc_colormtx_gain_t;
```

列挙定数	値	説明
VDC_COLORMTX_GAIN_GG	0	Y/G 信号出力の Y/G 信号のゲイン
VDC_COLORMTX_GAIN_GB	1	Y/G 信号出力の Cb/B 信号のゲイン
VDC_COLORMTX_GAIN_GR	2	Y/G 信号出力の Cr/R 信号のゲイン
VDC_COLORMTX_GAIN_BG	3	Cb/B 信号出力の Y/G 信号ゲイン
VDC_COLORMTX_GAIN_BB	4	Cb/B 信号出力の Cb/B 信号ゲイン
VDC_COLORMTX_GAIN_BR	5	Cb/B 信号出力の Cr/R 信号ゲイン
VDC_COLORMTX_GAIN_RG	6	Cr/R 信号出力の Y/G 信号ゲイン
VDC_COLORMTX_GAIN_RB	7	Cr/R 信号出力の Cb/B 信号ゲイン
VDC_COLORMTX_GAIN_RR	8	Cr/R 信号出力の Cr/R 信号ゲイン
VDC_COLORMTX_GAIN_NUM	9	カラーマトリクスのゲインパラメータ数

6.16 R_VDC_ImageEnhancement

概要	画質改善設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ImageEnhancement(const vdc_channel_t ch, const vdc_imgimprv_id_t imgimprv_id, const vdc_onoff_t shp_h_on, const vdc_enhance_sharp_t * const sharp_param, const vdc_onoff_t lti_h_on, const vdc_enhance_lti_t * const lti_param, const vdc_period_rect_t * const enh_area);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_imgimprv_id_t imgimprv_id: 画質改善部 ID — VDC_IMG_IMPRV_0: 画質改善部 0 本ドライバでは必ず画質改善部 0 を指定してください。 • vdc_onoff_t shp_h_on: シャープネス ON/OFF 設定 • vdc_enhance_sharp_t * sharp_param: シャープネス設定パラメータ NULL が指定された場合、設定は変更されません。 • vdc_onoff_t lti_h_on: LTI ON/OFF 設定 • vdc_enhance_lti_t * lti_param: LTI 設定パラメータ NULL が指定された場合、設定は変更されません。 • vdc_period_rect_t * enh_area: 画質改善領域設定パラメータ NULL が指定された場合、設定は変更されません。 上記の各パラメータがハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については構造体の説明を参照ください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_IF_CONDITION: インタフェース条件エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　**機能**

本関数では、画質改善に関する以下の処理を行います。

- シャープネスの ON/OFF 設定
- シャープネスのパラメータ設定
- LTI の ON/OFF 設定
- LTI のパラメータ設定
- シャープネスと LTI の適用される矩形領域設定

シャープネスや LTI は、パラメータの設定と ON/OFF の制御を別個に設定できます。一度設定されたパラメータの設定は、ハードウェアのリセットか、本関数による別の設定で上書きされるまでは有効です。

(2)　**使用条件**

本関数の使用時には、レイヤ 0 の読み出しプロセスが有効である必要があります。レイヤ 0 に対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤ 0 が有効になります。レイヤ 0 が無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

また、レイヤ 0 のカラーフォーマット (フレームバッファ読み出し信号のフォーマット)が YCbCr422 か YCbCr444 以外の場合、本処理は禁止です。この場合、インタフェース条件エラー (VDC_ERR_IF_CONDITION)を返します。

(3)　**パラメータ詳細**

vdc_enhance_sharp_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_onoff_t           shp_h2_lpf_sel;
    vdc_sharpness_ctrl_t  hrz_sharp[VDC_IMGENH_SHARP_NUM];
} vdc_enhance_sharp_t;
```

型 メンバ名	初期値	説明
vdc_onoff_t shp_h2_lpf_sel	VDC_OFF (0)	H2 エッジ検出前の折り返し除去用 LPF 選択 • VDC_OFF: LPF なし • VDC_ON: LPF あり
vdc_sharpness_ctrl_t hrz_sharp [VDC_IMGENH_SHARP_NUM]	-	シャープネス制御パラメータ 水平シャープネス (H1、H2、H3)

vdc_img_enh_sh_t はシャープネス帯域を表す列挙型です。

```
typedef enum
{
    VDC_IMGENH_SHARP_H1 = 0,
    VDC_IMGENH_SHARP_H2,
    VDC_IMGENH_SHARP_H3,
    VDC_IMGENH_SHARP_NUM
} vdc_img_enh_sh_t;
```

列挙定数	値	説明
VDC_IMGENH_SHARP_H1	0	水平シャープネス (H1)
VDC_IMGENH_SHARP_H2	1	水平シャープネス (H2)
VDC_IMGENH_SHARP_H3	2	水平シャープネス (H3)
VDC_IMGENH_SHARP_NUM	3	水平シャープネス帯域数

vdc_sharpness_ctrl_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint8_t    shp_clip_o;
    uint8_t    shp_clip_u;
    uint8_t    shp_gain_o;
    uint8_t    shp_gain_u;
    uint8_t    shp_core;
} vdc_sharpness_ctrl_t;
```

型 メンバ名	初期値	説明
uint8_t shp_clip_o	0	シャープネスの補正値クリップ (オーバーシュート側) 0x0000 ~ 0x00FF
uint8_t shp_clip_u	0	シャープネスの補正値クリップ (アンダーシュート側) 0x0000 ~ 0x00FF
uint8_t shp_gain_o	0	シャープネスのエッジ振幅値に対するゲイン設定 (オーバーシュート側) 0x0000 (0 倍) ~ 0x0040 (1 倍) ~ 0x00FF (約 4 倍)
uint8_t shp_gain_u	0	シャープネスのエッジ振幅値に対するゲイン設定 (アンダーシュート側) 0x0000 (0 倍) ~ 0x0040 (1 倍) ~ 0x00FF (約 4 倍)
uint8_t shp_core	0	シャープネスの能動範囲の指定 0x0000 ~ 0x007F

vdc_enhance_lti_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_onoff_t           lti_h2_lpf_sel;
    vdc_lti_mdfil_sel_t    lti_h4_median_tap_sel;
    vdc_lti_ctrl_t        lti[VDC_IMGENH_LTI_NUM];
} vdc_enhance_lti_t;
```

型 メンバ名	初期値	説明
vdc_onoff_t lti_h2_lpf_sel	VDC_OFF (0)	H2 エッジ検出前の折り返し除去用 LPF 選択 • VDC_OFF: LPF なし • VDC_ON: LPF あり
vdc_lti_mdfil_sel_t lti_h4_median_tap_sel	0	メディアンフィルタの参照画素選択 • VDC_LTI_MDFIL_SEL_ADJ2 (0): 隣接 2 画素目参照 • VDC_LTI_MDFIL_SEL_ADJ1 (1): 隣接 1 画素目参照
vdc_lti_ctrl_t lti[VDC_IMGENH_LTI_NUM]	-	LTI 制御パラメータ 水平 LTI (H2、H4)

vdc_img_enh_lti_t は LTI 帯域を表す列挙型です。

```
typedef enum
{
    VDC_IMGENH_LTI1 = 0,
    VDC_IMGENH_LTI2,
    VDC_IMGENH_LTI_NUM
} vdc_img_enh_lti_t;
```

列挙定数	値	説明
VDC_IMGENH_LTI1	0	H2、エッジ検出時に隣接 2 画素目参照
VDC_IMGENH_LTI2	1	H4、エッジ検出時に隣接 4 画素目参照
VDC_IMGENH_LTI_NUM	2	LTI 帯域数

vdc_lti_ctrl_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint8_t    lti_inc_zero;
    uint8_t    lti_gain;
    uint8_t    lti_core;
} vdc_lti_ctrl_t;
```

型 メンバ名	初期値	説明
uint8_t lti_inc_zero	10	メディアンフィルタの LTI 補正スレッシュ設定 0x0000 ~ 0x00FF
uint8_t lti_gain	0	LTI のエッジ振幅値に対するゲイン設定 0x0000 (0 倍) ~ 0x0040 (1 倍) ~ 0x00FF (約 4 倍)
uint8_t lti_core	0	LTI のコアリング 0x0000 ~ 0x00FF

vdc_period_rect_t 構造体については「5.3(1)」も参照ください。

型 メンバ名	初期値	説明
uint16_t vs	0	エンハンサ有効領域の垂直有効画像領域の開始位置設定 (ライン数) 2 ライン以上の設定にしてください。
uint16_t vw	0	エンハンサ有効領域の垂直有効画像領域の幅設定 (ライン数)
uint16_t hs	0	エンハンサ有効領域の水平有効画像領域の開始位置設定 (クロック数) 4 クロック以上の設定にしてください。
uint16_t hw	0	エンハンサ有効領域の水平有効画像領域の幅設定 (クロック数)

6.17 R_VDC_ImageBlackStretch

概要	黒伸張設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_ImageBlackStretch(const vdc_channel_t ch, const vdc_imgimprv_id_t imgimprv_id, const vdc_onoff_t bkstr_on, const vdc_black_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_imgimprv_id_t imgimprv_id: 画質改善部 ID — VDC_IMG_IMPRV_0: 画質改善部 0 本ドライバでは必ず画質改善部 0 を指定してください。 • vdc_onoff_t bkstr_on: 黒伸張 ON/OFF 設定 • vdc_black_t * param: 黒伸張設定パラメータ NULL が指定された場合、設定は変更されません。ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については構造体の説明を参照ください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_IF_CONDITION: インタフェース条件エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1) 機能

本関数では、指定された画質改善部の黒伸張に関する以下の処理を行います。

- 黒伸張処理の ON/OFF 設定
- 黒伸張処理のパラメータ設定

黒伸張処理は、黒伸張の設定と ON/OFF の制御を別個に設定できます。本関数による設定は、ハードウェアのリセットか、本関数による別の設定で上書きされるまでは有効です。

(2) 使用条件

本関数の使用時には、レイヤ 0 の読み出しプロセスが有効である必要があります。レイヤ 0 に対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤ 0 が有効になります。レイヤ 0 が無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

また、レイヤ 0 のカラーフォーマット (フレームバッファ読み出し信号のフォーマット)が YCbCr422 か YCbCr444 以外の場合、本処理は禁止です。この場合、インタフェース条件エラー (VDC_ERR_IF_CONDITION)を返します。

(3) パラメータ詳細

vdc_black_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t    bkstr_st;
    uint16_t    bkstr_d;
    uint16_t    bkstr_t1;
    uint16_t    bkstr_t2;
} vdc_black_t;
```

型 メンバ名	初期値	説明
uint16_t bkstr_st	0	黒伸張の開始点指定 0 (低)～15 (高)
uint16_t bkstr_d	0	黒伸張の深さ 0 (浅)～15 (深)
uint16_t bkstr_t1	0	黒伸張の時定数 (T1) 0 (小) ~ 31 (大)
uint16_t bkstr_t2	0	黒伸張の時定数 (T2) 0 (小) ~ 30 (大)

6.18　R_VDC_AlphaBlending

概要	アルファブレンディング設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_AlphaBlending(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_alpha_blending_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_alpha_blending_t * param: アルファブレンディング設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数では、矩形領域アルファブレンディング以外のアルファブレンディングに関する以下の処理を行います。

- ARGB1555/RGBA5551 フォーマットのα値設定
- 画素単位アルファブレンド時の、プレマルチプルド処理の設定

本関数では、レイヤ 2 と 3 の読み出し処理に対して ARGB1555/RGBA5551 のα値を設定できます。レイヤ 0 の読み出しで使用される ARGB1555/RGBA5551 のα値は、ドライバが自動的に'255 (= 1.0, 非透過)'を適用します。

(2)　使用条件

本関数の使用時には、layer_id にて指定されたレイヤが有効である必要があります。同じレイヤに対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。layer_id にて指定されたレイヤが無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

(3) パラメータ詳細

vdc_alpha_blending_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    const vdc_alpha_argb1555_t   * alpha_1bit;
    const vdc_alpha_pixel_t      * alpha_pixel;
} vdc_alpha_blending_t;
```

型 メンバ名	説明
const vdc_alpha_argb1555_t * alpha_1bit	ARGB1555/RGBA5551 フォーマットのα信号 NULL が指定された場合、設定は変更されません。
const vdc_alpha_pixel_t * alpha_pixel	画素単位アルファブレンド処理時の、プレマルチプルド処理 NULL が指定された場合、設定は変更されません。

alpha_1bit、alpha_pixel の各パラメータについて、ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については以下の各構造体の説明を参照ください。

vdc_alpha_argb1555_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint8_t    gr_a0;
    uint8_t    gr_a1;
} vdc_alpha_argb1555_t;
```

型 メンバ名	初期値	説明
uint8_t gr_a0	0	αが'0'の時のα信号 0 ~ 255
uint8_t gr_a1	0	αが'1'の時のα信号 0 ~ 255

vdc_alpha_pixel_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_onoff_t    gr_acalc_md;
} vdc_alpha_pixel_t;
```

型 メンバ名	初期値	説明
vdc_onoff_t gr_acalc_md	VDC_OFF (0)	画素単位アルファブレンド時の、プレマルチプルド処理 • VDC_OFF • VDC_ON

6.19　R_VDC_AlphaBlendingRect

概要	矩形領域アルファブレンディング設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_AlphaBlendingRect(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_onoff_t gr_arc_on, const vdc_alpha_blending_rect_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_onoff_t gr_arc_on: 矩形領域アルファブレンディング ON/OFF 設定 • vdc_alpha_blending_rect_t * param: 矩形領域アルファブレンディング設定パラメータ NULL が指定された場合、設定は変更されません。初期値については構造体の説明を参照ください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数では、矩形領域アルファブレンディングに関する以下の処理を行います。

- 矩形領域アルファブレンディングの ON/OFF 設定
- 矩形領域アルファブレンディングが適用される矩形領域設定
- 矩形領域アルファブレンディングに適用される α 値設定
- 矩形領域アルファブレンディングに適用されるフェードイン／フェードアウト設定

矩形領域のアルファブレンディングは、アルファブレンディングの設定と ON/OFF の制御を別個に設定できます。一度設定されたアルファブレンディングの設定は、ハードウェアのリセット、別の設定での上書き、若しくは指定されたレイヤリソースが関数「R_VDC_ReleaseDataControl」により破棄されるまでは有効です。

(2) 使用条件

本関数の使用時には、指定されたレイヤが有効である必要があります。同じレイヤに対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。layer_id にて指定されたレイヤが無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

(3) パラメータ詳細

vdc_alpha_blending_rect_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    const vdc_pd_disp_rect_t * gr_arc;
    const vdc_alpha_rect_t   * alpha_rect;
    const vdc_scl_und_sel_t  * scl_und_sel;
} vdc_alpha_blending_rect_t;
```

型 メンバ名	説明
const vdc_pd_disp_rect_t * gr_arc	矩形領域アルファブレンド領域設定 NULL が指定された場合、設定は変更されません。
const vdc_alpha_rect_t * alpha_rect	矩形領域アルファブレンドパラメータ NULL が指定された場合、設定は変更されません。
const vdc_scl_und_sel_t * scl_und_sel	スケーリング部の下層面の指定 このパラメータは本ドライバでは参照されません。 NULL を設定してください。

gr_arc のパラメータは API 関数「R_VDC_ReadDataControl」呼び出し時にドライバによりグラフィックス画像領域と同じ領域に初期化されます。alpha_rect について、ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については以下の構造体の説明を参照ください。gr_arc、alpha_rect の各パラメータを含む vdc_alpha_blending_rect_t 構造体について、ハードウェアリセット後に一度も設定が行われていない場合は、各パラメータの初期値のままとなります。

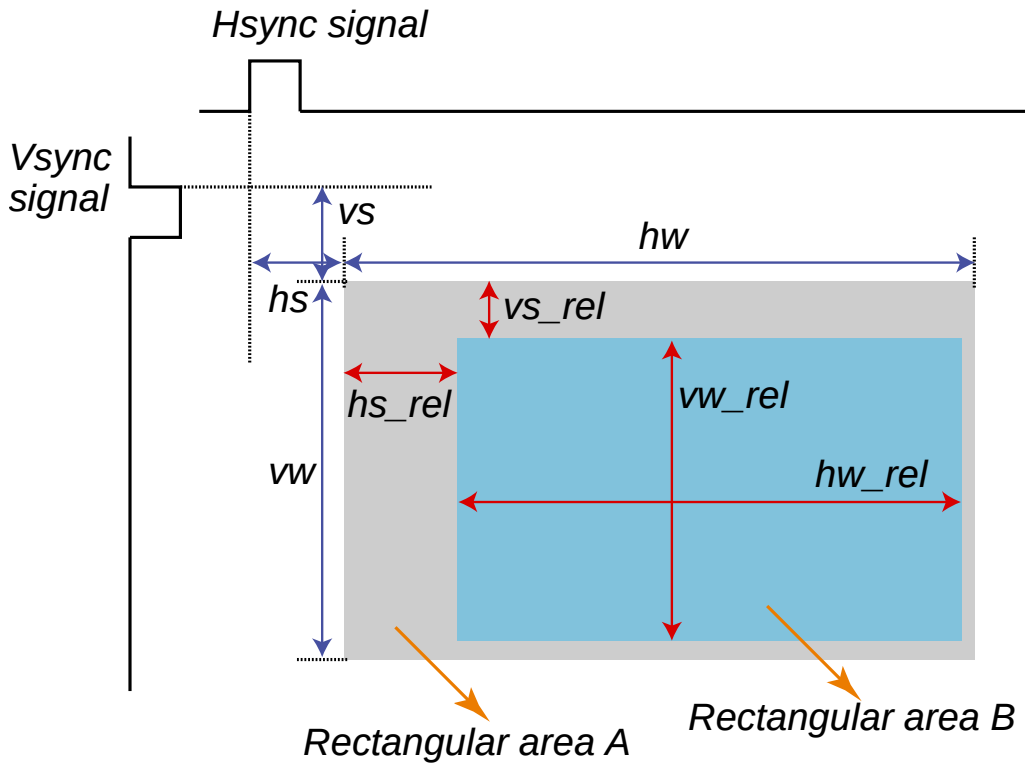
vdc_pd_disp_rect_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t   vs_rel;
    uint16_t   vw_rel;
    uint16_t   hs_rel;
    uint16_t   hw_rel;
} vdc_pd_disp_rect_t;
```

型 メンバ名	初期値	説明
uint16_t vs_rel	0	矩形領域アルファブレンド処理の有効画像領域の垂直信号開始位置 (ライン数) グラフィックス画像領域の垂直開始位置からの相対位置です。
uint16_t vw_rel	- ※	矩形領域アルファブレンド処理の有効画像領域の垂直幅 (ライン数)
uint16_t hs_rel	0	矩形領域アルファブレンド処理の有効画像領域の水平信号開始位置 (クロック数) グラフィックス画像領域の水平開始位置からの相対位置です。
uint16_t hw_rel	- ※	矩形領域アルファブレンド処理の有効画像領域の水平幅 (クロック数)

【注】 有効画像領域の垂直幅と水平幅は、ドライバによりグラフィックス画像領域と同じ値に初期化されます。

矩形領域アルファブレンディングの適用される矩形領域は、関数「R_VDC_ReadDataControl」で設定した該当レイヤのグラフィックス表示領域内における相対位置で指定されます(図 6-6 参照)。



【注】 Area A: グラフィックス表示領域
詳細は「6.9(3)」や「5.3(1)」を参照ください。
Area B: アルファブレンディング矩形領域

図 6-6　矩形領域アルファブレンディングの矩形領域

グラフィックス表示領域が関数「R_VDC_ChangeReadProcess」により変更された場合、アルファブレンディングの矩形領域は追従しません。矩形領域アルファブレンディングを使用時にグラフィックス表示領域を変更する場合は、アルファブレンディングの矩形領域も変更する必要があります。

vdc_alpha_rect_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    int16_t        gr_arc_coef;
    uint8_t        gr_arc_rate;
    uint8_t        gr_arc_def;
    vdc_onoff_t    gr_arc_mul;
} vdc_alpha_rect_t;
```

型 メンバ名	初期値	説明
int16_t gr_arc_coef	0	矩形領域アルファブレンド処理のアルファ係数 変化量 (-255 ~ +255)
uint8_t gr_arc_rate	0	矩形領域アルファブレンド処理のフレームレート Vsync が gr_arc_rate + 1 と同じ回数立ち上がるたびに、gr_arc_coef をα値に加算します。 0 ~ 255

uint8_t gr_arc_def	255	矩形領域アルファブレンド処理のアルファ初期値 0 ~ 255
vdc_onoff_t gr_arc_mul	VDC_OFF (0)	矩形領域アルファブレンド時の、カレントαとのマルチプライ処理 • VDC_OFF • VDC_ON

6.20　R_VDC_Chromakey

概要	クロマキー設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_Chromakey(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_onoff_t gr_ck_on, const vdc_chromakey_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_onoff_t gr_ck_on: クロマキーON/OFF 設定 • vdc_chromakey_t * param: クロマキー設定パラメータ NULL が指定された場合、設定は変更されません。ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については構造体の説明を参照ください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_IF_CONDITION: インタフェース条件エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　機能

本関数では、クロマキーに関する以下の処理を行います。

- クロマキーの ON/OFF 設定
- クロマキー対象の色信号、及び置換後の色信号設定

クロマキー処理は、クロマキーの設定と ON/OFF の制御を別個に設定できます。一度設定されたクロマキーの設定は、ハードウェアのリセット、別の設定での上書き、若しくは指定されたレイヤリソースが関数「R_VDC_ReleaseDataControl」により破棄されるまでは有効です。

(2)　使用条件

本関数の使用時には、layer_id にて指定されたレイヤが有効である必要があります。同じレイヤに対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。layer_id にて指定されたレイヤが無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

また、指定されているレイヤのカラーフォーマット (フレームバッファ読み出し信号のフォーマット)が YCbCr422 か YCbCr444 の場合、クロマキー処理は禁止です。この場合、インタフェース条件エラー (VDC_ERR_IF_CONDITION)を返します。

(3)　パラメータ詳細

vdc_chromakey_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint32_t    ck_color;
    uint32_t    rep_color;
    uint8_t     rep_alpha;
} vdc_chromakey_t;
```

型 メンバ名	初期値	説明
uint32_t ck_color	0	RGB/CLUT 参照クロマキー処理対象 RGB/CLUT 信号 ターゲットとなるレイヤで使用されているカラーフォーマットで指定して下さい(LSB 詰め)。
uint32_t rep_color	0	RGB/CLUT 参照クロマキー処理置換後 RGB 信号 ターゲットとなるレイヤで使用されているカラーフォーマットで指定して下さい(LSB 詰め)。ただし、カラーフォーマットが CLUT8、CLUT4、CLUT1 の場合は RGB888 フォーマットで指定します。 ここで指定されたアルファ値は無視されます。クロマキー処理置換後のアルファ値は rep_alpha にて指定してください。
uint8_t rep_alpha	0 ※	RGB/CLUT 参照クロマキー処理置換後アルファ信号 アルファ値を 8 ビットで指定して下さい。 0 ~ 255

【注】 レイヤ 0 の場合のα値は、ドライバで自動的に'255'が設定されます。

6.21　R_VDC_CLUT

概要	CLUT 設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_CLUT(const vdc_channel_t ch, const vdc_layer_id_t layer_id, const vdc_clut_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0　読み出し処理 — VDC_LAYER_ID_2_RD: レイヤ 2　読み出し処理 — VDC_LAYER_ID_3_RD: レイヤ 3　読み出し処理 • vdc_clut_t * param: CLUT 設定パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_EXCEED_RANGE: 設定範囲外エラー — VDC_ERR_RESOURCE_LAYER: レイヤリソースエラー

詳細

(1)　**機能**

本関数では、指定されたレイヤの CLUT の設定を行います。

(2)　**使用条件**

本関数の使用時には、layer_id にて指定されたレイヤが有効である必要があります。同じレイヤに対して関数「R_VDC_ReadDataControl」を呼び出すことで、レイヤが有効になります。layer_id にて指定されたレイヤが無効な場合、レイヤリソースエラー (VDC_ERR_RESOURCE_LAYER)を返します。

(3) パラメータ詳細

vdc_clut_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint32_t          color_num;
    const uint32_t    * clut;
} vdc_clut_t;
```

型 メンバ名	説明
uint32_t color_num	CLUT の色の数 CLUT1 形式使用時: 1 ~ 2 CLUT4 形式使用時: 1 ~ 16 CLUT8 形式使用時: 1 ~ 256
const uint32_t * clut	CLUT データ(ARGB8888 形式)格納アドレス NULL を設定しないでください。

6.22 R_VDC_DisplayCalibration

概要	画面出力較正処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_DisplayCalibration(const vdc_channel_t ch, const vdc_disp_calibration_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_disp_calibration_t * param: 画面出力較正パラメータ NULL は設定しないでください。
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_NULL: NULL 指定エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_PARAM_UNDEFINED: 未定義パラメータ指定エラー — VDC_ERR_RESOURCE_OUTPUT: 出力リソースエラー

詳細

(1) 機能

本関数では、画面出力較正に関する以下の処理を行います。

- パネルブライト設定
- コントラスト調整設定
- パネルディザ設定
- パネル出力補正回路の制御設定

本関数による設定は、ハードウェアのリセットか、本関数による別の設定で上書きされるまでは有効です。

(2) 使用条件

本関数の使用前に、関数「R_VDC_DisplayOutput」を呼び出すことで、ディスプレイ出力を有効にする必要があります。ディスプレイ出力が無効な場合、出力リソースエラー (VDC_ERR_RESOURCE_OUTPUT)を返します。

(3) パラメータ詳細

vdc_disp_calibration_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_calibr_route_t           route;
    const vdc_calibr_bright_t    * bright;
    const vdc_calibr_contrast_t  * contrast;
    const vdc_calibr_dither_t    * panel_dither;
} vdc_disp_calibration_t;
```

型 メンバ名	説明
vdc_calibr_route_t route	補正回路の順番の制御 • VDC_CALIBR_ROUTE_BCG: ブライト ⇒ コントラスト ⇒ ガンマ補正 • VDC_CALIBR_ROUTE_GBC: ガンマ補正 ⇒ ブライト ⇒ コントラスト
const vdc_calibr_bright_t * bright	ブライト(DC)調整パラメータ 変更する必要が無い場合は NULL を指定してください。
const vdc_calibr_contrast_t * contrast	コントラスト(ゲイン)調整パラメータ 変更する必要が無い場合は NULL を指定してください。
const vdc_calibr_dither_t * panel_dither	パネルディザパラメータ 変更する必要が無い場合は NULL を指定してください。

brigte、contrast、panel_dither の各パラメータについて、ハードウェアリセット後に一度も設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値については以下の各構造体の説明を参照ください。

vdc_calibr_bright_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint16_t    pbrt_g;
    uint16_t    pbrt_b;
    uint16_t    pbrt_r;
} vdc_calibr_bright_t;
```

型 メンバ名	初期値	説明
uint16_t pbrt_g	512	G 信号のブライト(DC)調整 0x0000 (-512) ~ 0x03FF (+511)
uint16_t pbrt_b	512	B 信号のブライト(DC)調整 0x0000 (-512) ~ 0x03FF (+511)
uint16_t pbrt_r	512	R 信号のブライト(DC)調整 0x0000 (-512) ~ 0x03FF (+511)

vdc_calibr_contrast_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    uint8_t    cont_g;
    uint8_t    cont_b;
    uint8_t    cont_r;
} vdc_calibr_contrast_t;
```

型 メンバ名	初期値	説明
uint8_t cont_g	128	G 信号のコントラスト(ゲイン)調整 0x0000 (0/128[倍]) ~ 0x00FF (255/128[倍])
uint8_t cont_b	128	B 信号のコントラスト(ゲイン)調整 0x0000 (0/128[倍]) ~ 0x00FF (255/128[倍])
uint8_t cont_r	128	R 信号のコントラスト(ゲイン)調整 0x0000 (0/128[倍]) ~ 0x00FF (255/128[倍])

vdc_calibr_dither_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    vdc_panel_dither_md_t  pdth_sel;
    uint8_t                pdth_pa;
    uint8_t                pdth_pb;
    uint8_t                pdth_pc;
    uint8_t                pdth_pd;
} vdc_calibr_dither_t;
```

型 メンバ名	初期値	説明
vdc_panel_ dither_md_t pdth_sel	0	パネルディザ動作モード • VDC_PDTH_MD_TRU (0): 切り捨て • VDC_PDTH_MD_RDOF (1): 四捨五入 • VDC_PDTH_MD_2X2 (2): 2x2 パターンディザ • VDC_PDTH_MD_RAND (3): ランダムパターンディザ
uint8_t pdth_pa	3	2x2 パターンディザのパターン値(A) 0 ~ 3 pdth_sel に VDC_PDTH_MD_2X2 が指定された場合のみ参照されます。
uint8_t pdth_pb	0	2x2 パターンディザのパターン値(B) 0 ~ 3 pdth_sel に VDC_PDTH_MD_2X2 が指定された場合のみ参照されます。
uint8_t pdth_pc	2	2x2 パターンディザのパターン値(C) 0 ~ 3 pdth_sel に VDC_PDTH_MD_2X2 が指定された場合のみ参照されます。
uint8_t pdth_pd	1	2x2 パターンディザのパターン値(D) 0 ~ 3 pdth_sel に VDC_PDTH_MD_2X2 が指定された場合のみ参照されます。

6.23 R_VDC_GammaCorrection

概要	ガンマ補正設定処理
ヘッダ	r_vdc.h
宣言	vdc_error_t R_VDC_GammaCorrection(const vdc_channel_t ch, const vdc_onoff_t gam_on, const vdc_gamma_correction_t * const param);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_onoff_t gam_on: ガンマ補正 ON/OFF 設定 • vdc_gamma_correction_t * param: ガンマ補正設定パラメータ
リターン値	• vdc_error_t: エラーコード — VDC_OK: 正常終了 — VDC_ERR_PARAM_CHANNEL: チャネル不正エラー — VDC_ERR_PARAM_BIT_WIDTH: ビット幅エラー — VDC_ERR_RESOURCE_OUTPUT: 出力リソースエラー

詳細

(1) 機能

本関数では、ガンマ補正に関する以下の処理を行います。

- ガンマ補正の ON/OFF 設定
- G/B/R 信号のガンマ補正ゲイン調整値設定
- G/B/R 信号のガンマ補正開始閾値設定

ガンマ補正処理は、ガンマ補正のパラメータ設定と ON/OFF の制御を別個に設定できます。一度設定されたガンマ補正のパラメータは、ハードウェアのリセットか、別の設定で上書きされるまでは有効です。

(2) 使用条件

本関数の使用前に、関数「R_VDC_DisplayOutput」を呼び出すことで、ディスプレイ出力を有効にする必要があります。ディスプレイ出力が無効な場合、出力リソースエラー (VDC_ERR_RESOURCE_OUTPUT)を返します。

(3)　パラメータ詳細

vdc_gamma_correction_t 構造体のメンバは以下の通りです。

```
typedef struct
{
    const uint16_t    * gam_g_gain;
    const uint8_t     * gam_g_th;
    const uint16_t    * gam_b_gain;
    const uint8_t     * gam_b_th;
    const uint16_t    * gam_r_gain;
    const uint8_t     * gam_r_th;
} vdc_gamma_correction_t;
```

型 メンバ名	説明
const uint16_t * gam_g_gain	G 信号の領域 0 ~ 31 のゲイン調整 符号無し (0 ~ 2047[LSB], 1024[LSB] = 1.0[倍]) 変更する必要が無い場合は NULL を指定してください。
const uint8_t * gam_g_th	G 信号の領域 1 ~ 31 の開始閾値 符号無し (0 ~ 255[LSB]) 変更する必要が無い場合は NULL を指定してください。
const uint16_t * gam_b_gain	B 信号の領域 0 ~ 31 のゲイン調整 符号無し (0 ~ 2047[LSB], 1024[LSB] = 1.0[倍]) 変更する必要が無い場合は NULL を指定してください。
const uint8_t * gam_b_th	B 信号の領域 1 ~ 31 の開始閾値 符号無し (0 ~ 255[LSB]) 変更する必要が無い場合は NULL を指定してください。
const uint16_t * gam_r_gain	R 信号の領域 0 ~ 31 のゲイン調整 符号無し (0 ~ 2047[LSB], 1024[LSB] = 1.0[倍]) 変更する必要が無い場合は NULL を指定してください。
const uint8_t * gam_r_th	R 信号の領域 1 ~ 31 の開始閾値 符号無し (0 ~ 255[LSB]) 変更する必要が無い場合は NULL を指定してください。

ハードウェアリセット後、一度もパラメータ設定が行われていない場合は、ハードウェアマニュアルに定められた初期値のままとなります。初期値は以下の通りです。

- G/B/R 信号の領域 0 ~ 31 のゲイン調整値: 全て 1024 (= 1.0[倍])
- G/B/R 信号の領域 1 ~ 31 の開始閾値: 領域 n の開始閾値は n x 8

6.24 R_VDC_GetISR

概要	割り込みサービスルーチン取得処理
ヘッダ	r_vdc.h
宣言	void (*R_VDC_GetISR(const vdc_channel_t ch, const vdc_int_type_t type)) (const uint32_t int_sense);
引数	• vdc_channel_t ch: チャネル — VDC_CHANNEL_0: チャネル 0 本ドライバでは必ずチャネル 0 を指定してください。 • vdc_int_type_t type: 割り込みタイプ 詳細は「5.2(8)」を参照してください。
リターン値	• void (*)(const uint32_t int_sense): 割り込みサービスルーチンの関数ポインタ — 0 以外: 正常終了 — 0: エラー

詳細

(1) 機能

本関数は、指定された割り込みのサービスルーチンの関数ポインタを返します。

ch にて指定されたチャネルや、type にて指定された割り込みタイプが不正であった場合、'0'を返します。

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

6.25 R_SPEA_WindowOffset

概要	Window の座標オフセットの設定
ヘッダ	r_spea.h
宣言	spea_error_t R_SPEA_WindowOffset(const vdc_layer_id_t layer_id, const uint16_t offset_x, const uint16_t offset_y);
引数	• vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_2_RD: レイヤ 2 — VDC_LAYER_ID_3_RD: レイヤ 3 VDC_LAYER_ID_0_RD は、設定しないでください。 • uint16_t offset_x: offset_x は、2[pixel]単位で 0 以上 2047 以下を設定してください。 • uint16_t offset_y: offset_y は、0 以上 8191 以下を設定してください。
リターン値	• spea_error_t: エラーコード — SPEA_OK: 正常終了 — SPEA_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — SPEA_ERR_PARAM: 不許可条件エラー

詳細

(1) 機能

本関数では、データ読み出し制御に関する以下の処理を行います。

- SPEA の仮想フレームに対する、VDC(レイヤ 2 及び 3)の表示領域の配置を設定する

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

(3)　パラメータ詳細

本関数では、図 6-7 に示すように SPEA の仮想フレームに対する VDC の表示領域の配置を設定します。このオフセットは、SPEA の Window 位置を設定する時の原点(0,0)となります。

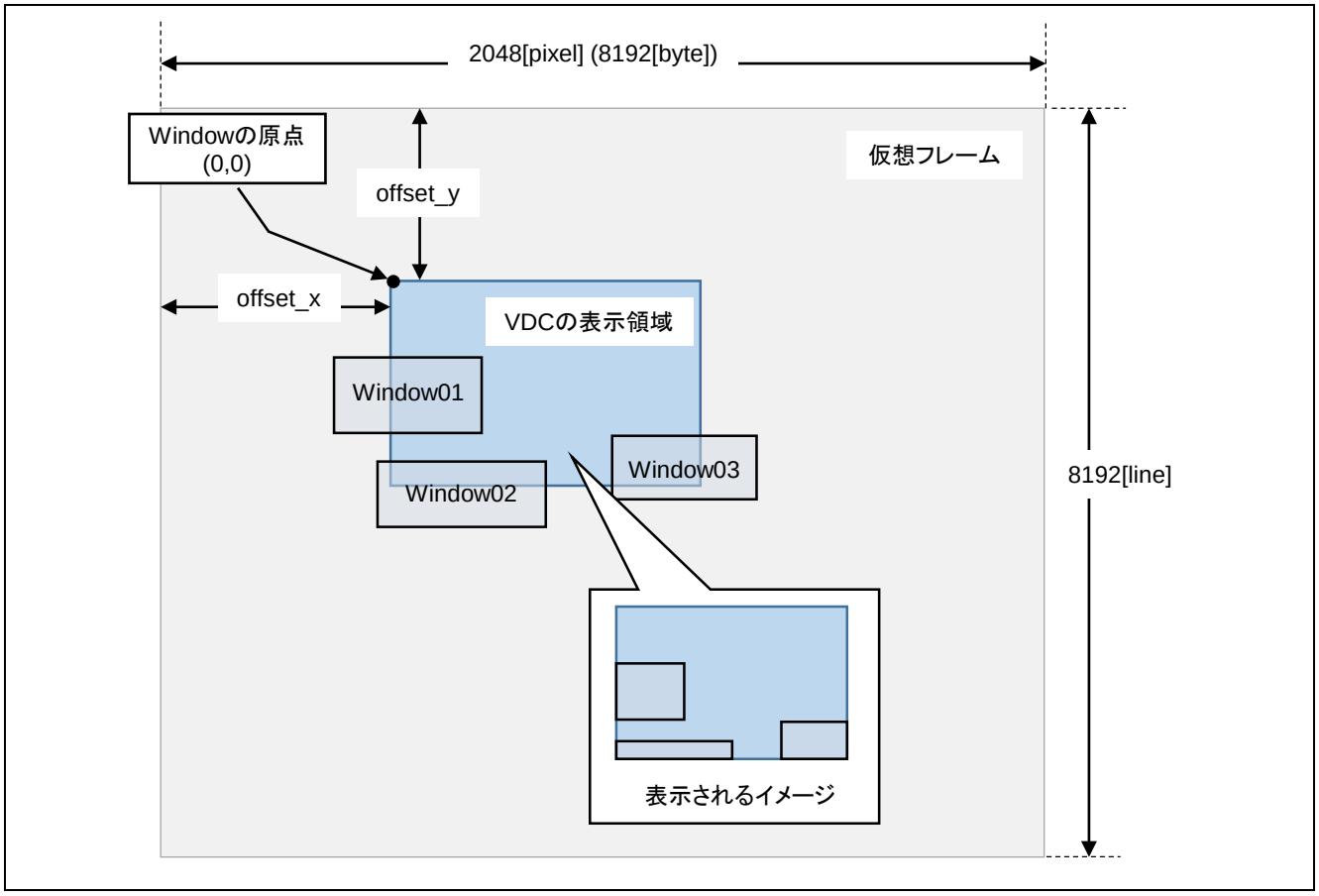


図 6-7　仮想フレームと VDC 表示領域の関係

6.26 R_SPEA_SetWindow

概要	Window パラメータの設定
ヘッダ	r_spea.h
宣言	spea_error_t R_SPEA_SetWindow(const vdc_layer_id_t layer_id, const spea_window_id_t window_id, const spea_onoff_t sken, const spea_sklym_t * sklym, const spea_skpsm_t * skpsm, const void * buffer);
引数	• vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_2_RD: レイヤ 2 — VDC_LAYER_ID_3_RD: レイヤ 3 VDC_LAYER_ID_0_RD は、設定しないでください。 • spea_window_id_t window_id: Window ID — WINDOW_00 – WINDOW_15: Window ID • spea_onoff_t sken: Window の ON/OFF — SPEA_ON : — SPEA_OFF : • spea_sklym_t * sklym: Window サイズ — sklym.width は、2[pixel]単位で 0 以上 2047 以下を設定してください。 — sklym.height は、0 以上 8191 以下を設定してください。 • spea_skpsm_t * skpsm: Window 開始座標 — skpsm.x は、2[pixel]単位で設定してください。また、R_SPEA_WindowOffset で設定した offset_x を skpsm.x に加算した結果が 0 以上 2047 以下でない場合エラーとなります。 — skpsm.y は、R_SPEA_WindowOffset で設定した offset_y を skpsm.y に加算した結果が 0 以上 8191 以下でない場合エラーとなります。 • void * buffer : Window の読み出しバッファアドレス — 8 バイトアライメントのアドレスを指定してください。
リターン値	• spea_error_t: エラーコード — SPEA_OK: 正常終了 — SPEA_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — SPEA_ERR_PARAM: 不許可条件エラー

詳細

(1) 機能

本関数では、データ読み出し制御に関する以下の処理を行います。

- SPEA の Window の表示/非表示
- SPEA の Window 開始座標、サイズ、読み出しバッファの設定
- VDC のフレームバッファバースト転送モードの設定(SPEA_ON:128 バイト SPEA_OFF:32 バイト転送)

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。SPEA は、VDC のレイヤ 2 や 3 を使用して動作します。その為、VDC の設定も行ってください。

(3) スプライトレイヤ

SPEA を使用する場合、VDC のレイヤ 2 やレイヤ 3 の設定が必要です。VDC のレイヤ設定に使用するフレームバッファのベースアドレスや、フレームバッファのラインオフセットアドレスには、以下の値を設定してください。

```
#define VIRTUAL_FRAME_BASE_ADD      (0x30000000u)
#define VIRTUAL_FRAME_STRAID        (8192u)
```

6.27 R_SPEA_WindowUpdate

概要	Window パラメータの更新要求
ヘッダ	r_spea.h
宣言	spea_error_t R_SPEA_WindowUpdate(const vdc_layer_id_t layer_id);
引数	• vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_2_RD: レイヤ 2 — VDC_LAYER_ID_3_RD: レイヤ 3 VDC_LAYER_ID_0_RD は、設定しないでください。
リターン値	• spea_error_t: エラーコード — SPEA_OK: 正常終了 — SPEA_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー

詳細

(1) 機能

本関数では、データ読み出し制御に関する以下の処理を行います。

- SPEA の Window パラメータの更新要求

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

6.28　R_RLE_SetWindow

概要	RLE パラメータの設定
ヘッダ	r_spea.h
宣言	spea_error_t R_RLE_SetWindow(const vdc_error_t layer_id, const rle_onoff_t sken, const rle_cfg_t * rle_cfg, const void * buffer)
引数	• vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0 VDC_LAYER_ID_2_RD、VDC_LAYER_ID_3_RD は、設定しないでください。 • rle_onoff_t sken: RLE の ON/OFF — RLE_ON : — RLE_OFF : • rle_cfg_t * rle_cfg: NULL を設定してください(TBD) • void * buffer : Window の読み出しバッファアドレス — 8 バイトアライメントのアドレスを指定してください。また、Targa 形式の画像ファイルのヘッダを取り除いたデータを設定してください。
リターン値	• spea_error_t: エラーコード — SPEA_OK: 正常終了 — SPEA_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー — SPEA_ERR_PARAM: 不許可条件エラー

詳細

(1)　機能

本関数では、データ読み出し制御に関する以下の処理を行います。

- SPEA の RLE の有効/無効
- SPEA の RLE パラメータの設定

(2)　使用条件

本関数の呼び出しに必要な条件は特にありません。VDC のレイヤ 0 と連動して動作する為、VDC のレイヤ 0 の設定も行ってください。

(3)　RLE(ランレングス符号化)レイヤ

SPEA の RLE ユニットは、VDC のレイヤ 0 と連動することが可能です。Targa 形式で圧縮されたデータを展開して表示することができます。これにより少ないメモリで高解像度の表示を実現することが可能です。但し、圧縮率は画像に依存します。

(4) Targa 形式の画像作成

Targa 形式の画像ファイル作成には、GIMP などのグラフィックスエディタツールをご使用ください。
(GIMP2.8 で動作確認)

(5) 画像作成の制限事項

RLE ユニットに使用する Targa 形式の画像では、以下の制限を守る必要があります。

【注】1. 水平方向の画像データサイズが 128 バイトアライメントに合わない場合、
ライン毎に 128 バイトアライメントとなるようにダミーデータを挿入する必要があります。
2. RLE ユニットは、各ラインの読み取り終了後に 128 バイトの追加読み取りを実行します。
このため、ライン毎に 128 バイトの追加のダミーデータを挿入する必要があります。
3. RLE 圧縮カラーデータフォーマットは、24 bpp(アルファチャンネルが含まれない)。

(例) 32bpp の水平 240 ピクセル画像において各ラインに必要となるダミーデータ挿入(図 6-8)
注意 1 の制限を満たすために 64 バイトのダミーデータが必要であり、注意 2 の制限を満たすためさらに 128 バイトのダミーデータが必要となります。合計で 192 バイトのダミーデータ挿入が必要となり、ダミーデータ挿入後の水平ピクセル数は 288 ピクセルとなります。

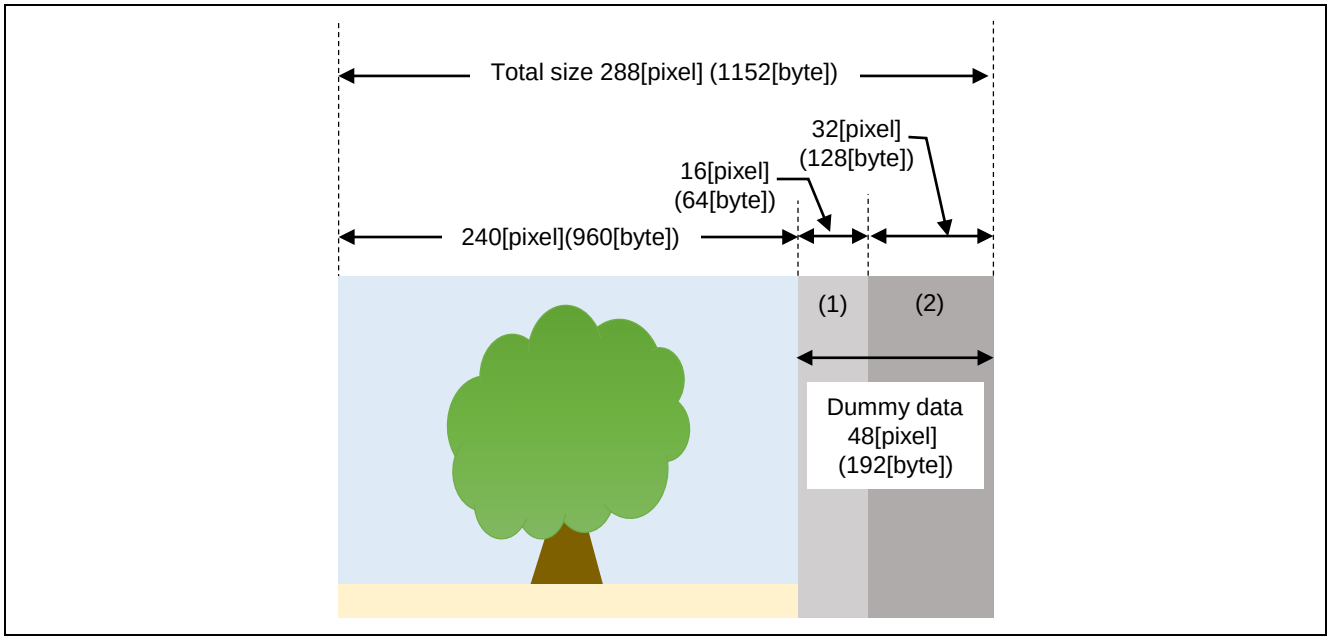


図 6-8　ダミーデータについて

(6)　Targa 形式の画像ファイルのヘッダ

ヘッダを削除した Targa 形式のイメージファイルを指定します。Targa 形式のヘッダは 18 [byte]に固定されています。ヘッダ以降のデータを使用してください。

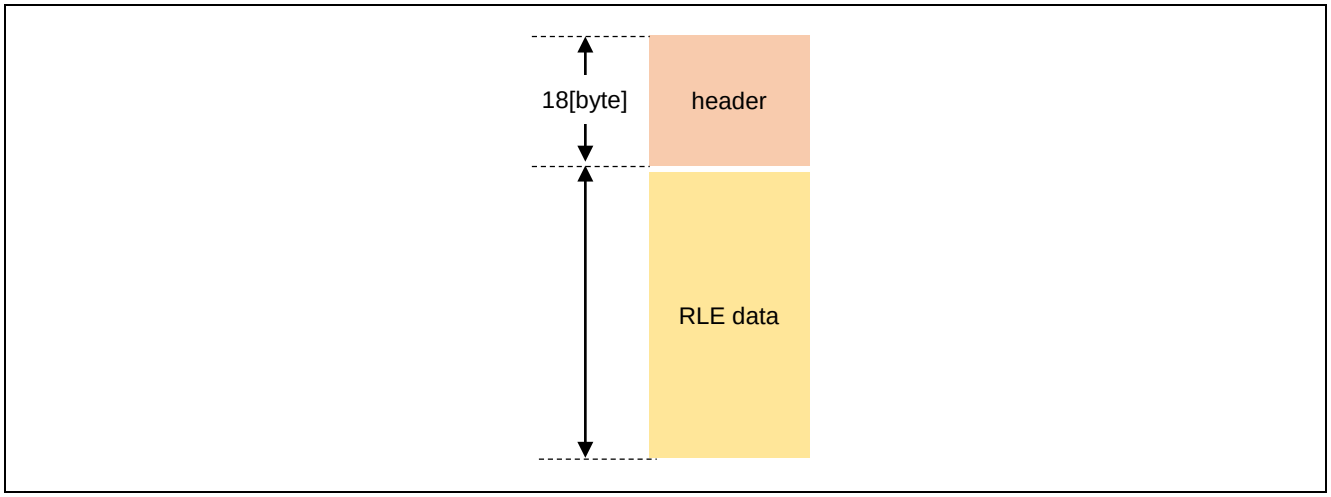


図 6-9　Targa 形式の画像ファイルのヘッダ

6.29 R_RLE_WindowUpdate

概要	RLE パラメータの更新要求
ヘッダ	r_spea.h
宣言	spea_error_t R_RLE_WindowUpdate(const vdc_layer_id_t layer_id);
引数	• vdc_layer_id_t layer_id: レイヤ ID — VDC_LAYER_ID_0_RD: レイヤ 0 VDC_LAYER_ID_2_RD、VDC_LAYER_ID_3_RD は、設定しないでください。
リターン値	• spea_error_t: エラーコード — SPEA_OK: 正常終了 — SPEA_ERR_PARAM_LAYER_ID: レイヤ ID 不正エラー

詳細

(1) 機能

本関数では、データ読み出し制御に関する以下の処理を行います。

- SPEA の RLE パラメータの更新要求

(2) 使用条件

本関数の呼び出しに必要な条件は特にありません。

7. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RZ/A2M グループ ユーザーズマニュアル ハードウェア編
（最新版をルネサス エレクトロニクスホームページから入手してください。）

RTX921053C00000BE（**RZ/A2M** CPU ボード）ユーザーズマニュアル
（最新版をルネサス エレクトロニクスホームページから入手してください。）

RTK79210XXB00000BE（**RZ/A2M** SUB ボード）ユーザーズマニュアル
（最新版をルネサス エレクトロニクスホームページから入手してください。）

Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition Issue C
（最新版を Arm ホームページから入手してください。）

Arm Cortex™-A9 Technical Reference Manual Revision: r4p1
（最新版を Arm ホームページから入手してください。）

Arm Generic Interrupt Controller Architecture Specification - Architecture version2.0
（最新版を Arm ホームページから入手してください。）

Arm CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual Revision: r3p3
（最新版を Arm ホームページから入手してください。）

テクニカルアップデート／テクニカルニュース
（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：統合開発

統合開発環境 e2 studio のユーザーズマニュアルは、ルネサス エレクトロニクスホームページから入手してください。
（最新版をルネサス エレクトロニクスホームページから入手してください。）

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/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2018.9.14	-	初版
1.01	2018.12.28	p.97	6.28 R_RLE_SetWindow 引数 rle_onoff_t に変更

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生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、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またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 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、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング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）、交通制御（信号）、大規模通信機器、
   金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment向け製品と定義しているものを除き、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（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合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、データシート等において高信頼性、Harsh environment向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他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。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. 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。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
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。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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直接的、間接的に支配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。

(Rev.4.0-1 2017.11)

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　〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

■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：https://www.renesas.com/contact/