Introduction

This application note describes the sample driver for the RZ/A1H video display controller 5 (VDC5) and a sample application that makes use of that driver.

Target Device

RZ/A1H Group

Contents

1. Introduction .................................................................................................................... 3
2. VDC5 Driver .................................................................................................................... 4
   2.1 Overview ..................................................................................................................... 4
   2.1.1 Functions .................................................................................................................. 4
   2.1.2 Environments .......................................................................................................... 5
   2.1.3 Memory Requirements ............................................................................................ 5
   2.1.4 File Configuration .................................................................................................... 5
   2.1.5 VDC5 Internal Module Configuration ................................................................. 7
   2.1.6 Cascaded Connection ............................................................................................... 8
   2.1.7 Restrictions ............................................................................................................. 9
   2.2 Common Definitions .................................................................................................. 10
   2.2.1 Constant Definitions ............................................................................................... 10
   2.2.2 Enumeration Type Definitions ............................................................................... 10
   2.2.3 Structure Definitions ............................................................................................... 15
   2.2.4 Error Codes ............................................................................................................. 17
   2.2.5 Compile Switch ....................................................................................................... 18
   2.2.6 User Custom Parameters ....................................................................................... 19
   2.3 API Functions .......................................................................................................... 21
   2.3.1 R_VDC5_Initialize .................................................................................................. 22
   2.3.2 R_VDC5_Terminate ................................................................................................. 26
   2.3.3 R_VDC5_VideoInput ............................................................................................... 27
   2.3.4 R_VDC5_SyncControl ............................................................................................ 31
   2.3.5 R_VDC5_DisplayOutput ......................................................................................... 35
   2.3.6 R_VDC5_CallbackISR .......................................................................................... 40
   2.3.7 R_VDC5_WriteDataControl ............................................................................... 42
   2.3.8 R_VDC5_ChangeWriteProcess ............................................................................. 47
   2.3.9 R_VDC5_ReadDataControl ................................................................................... 49
   2.3.10 R_VDC5_ChangeReadProcess ......................................................................... 54
   2.3.11 R_VDC5_StartProcess ....................................................................................... 56
   2.3.12 R_VDC5_StopProcess ......................................................................................... 59
2.3.13  R_VDC5_ReleaseDataControl ................................................................. 60  
2.3.14  R_VDC5_VideoNoiseReduction ............................................................... 62  
2.3.15  R_VDC5_ImageColorMatrix ................................................................. 64  
2.3.16  R_VDC5_ImageEnhancement ................................................................. 67  
2.3.17  R_VDC5_ImageBlackStretch ............................................................... 71  
2.3.18  R_VDC5_AlphaBlending ..................................................................... 73  
2.3.19  R_VDC5_AlphaBlendingRect .............................................................. 75  
2.3.20  R_VDC5_Chromakey ....................................................................... 79  
2.3.21  R_VDC5_CLUT .............................................................................. 81  
2.3.22  R_VDC5_DisplayCalibration .............................................................. 83  
2.3.23  R_VDC5_GammaCorrection ............................................................... 86  
2.3.24  R_VDC5_GetISR ........................................................................ 88  

3.  Sample Application ....................................................................................... 89  
3.1  Specifications ............................................................................................ 89  
3.2  Operation Check Conditions ....................................................................... 89  
3.3  Related Application Note .......................................................................... 90  
3.4  Description of Software ........................................................................... 90  
  3.4.1  System Outline ..................................................................................... 90  
  3.4.2  Memory Mappings ............................................................................... 91  
  3.4.3  Interrupts ............................................................................................... 91  
3.5  Software Details ......................................................................................... 92  
  3.5.1  List of Functions .................................................................................. 92  
  3.5.2  Function Specifications ...................................................................... 93  
  3.5.3  Flowcharts ........................................................................................ 102  
  3.5.4  LCD Setup ......................................................................................... 106  

4.  Sample Code .................................................................................................. 111  

5.  Related Documents ......................................................................................... 111
1. Introduction

This application note describes the sample driver for the RZ/A1H video display controller 5 (VDC5) and a sample application that makes use of that driver.
2. VDC5 Driver

2.1 Overview

2.1.1 Functions

The functions that this driver supports are listed below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
</tr>
</thead>
</table>
| Input video image specification | 8-bit input conforming to ITU-R BT.656 standard (27 MHz, interface signal)  
  8-bit input conforming to ITU-R BT.656 extended standard (27 MHz, progressive signal) *  
  8-bit input conforming to ITU-R BT.601 extended standard (27 MHz, interface signal) *  
  8-bit input conforming to ITU-R BT.601 extended standard (54 MHz, progressive signal) *  
  16-bit input conforming to ITU-R BT.601 extended standard (13.5 MHz, interface signal) *  
  Digital pin input: YCbCr422, YCbCr444, RGB888, RGB666, and RGB565 video image |
| Video image recording | Storing the video image in the YCbCr422/YCbCr444/RGB565/RGB888 format at a rate of 1/1, 1/2, 1/4, or 1/8 field. |
| Video image quality adjustment | Contrast adjustment, brightness adjustment, horizontal noise reduction, black stretch, LTI/sharpness |
| Video image scaling/rotation | Vertical/horizontal scaling: 1/8 to 8 times  
  0, 90, 180, and 270 degree rotations and horizontal mirroring |
| Graphics planes | Graphics planes: 4 planes  
  Supported pixel formats: RGB565, RGB888, ARGB1555, ARGB4444, ARGB8888, RGBA5551, RGBA8888, CLUT8, CLUT4, CLUT1, YCbCr422, YCbCr444 |
| Graphics functions | Alpha blending in rectangular area (fade-in and fade-out functions are available.)  
  Chroma-key  
  Alpha blending in one pixel units |
| Output video image size | Video output size examples: WXGA (1280x768), XGA (1024x768), SVGA (800x600), WVGA (800x480), VGA (640x480), WQVGA (480x240), QVGA landscape (320x240), QVGA portrait (240x320) |
| Output video image format | Progressive video output  
  • RGB888 (24-bit parallel output)  
  • RGB666 (18-bit parallel output)  
  • RGB565 (16-bit parallel output)  
  • RGB888 (8-bit serial output) |
| Panel output adjustment | Panel brightness/contrast adjustment, RGB gamma correction, dither processing, output format conversion |

Note: The ITU-R BT.656, 601 standard does not include the description regarding the progressive signal. The ITU-R BT.601 standard does not include the description regarding the connection interface.
2.1.2 Environments

The development and operating environments of this driver are summarized below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcomputer used</td>
<td>RZ/A1H, CPU: ARM9</td>
</tr>
<tr>
<td>Operating frequency [MHz]</td>
<td>CPU Clock: 400.0</td>
</tr>
<tr>
<td></td>
<td>Internal Bus Clock: 133.33</td>
</tr>
<tr>
<td></td>
<td>Peripheral Clock 1: 66.67</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>Source voltage (I/O): 3.3</td>
</tr>
<tr>
<td></td>
<td>Source voltage (internal): 1.18</td>
</tr>
<tr>
<td>Development environment</td>
<td>ARM Development Studio 5 (DS-5™) Version 5.16</td>
</tr>
<tr>
<td>Compiler</td>
<td>ARM C/C++ Compiler/Linker/Assembler, 5.03 [Build 102]</td>
</tr>
<tr>
<td>Board to be used</td>
<td>R7S72100 CPU Board (part number: RTK772100BC00000BR)</td>
</tr>
<tr>
<td></td>
<td>R7S72100 Optional Board (part number: RTK772100B00000BR)</td>
</tr>
</tbody>
</table>

2.1.3 Memory Requirements

Table 2-3 shows required memory sizes.

Table 2-3 Memory Sizes

<table>
<thead>
<tr>
<th>Memory used</th>
<th>Size [Kbyte]</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td>31.72</td>
<td>Code, constant data, initialization data</td>
</tr>
<tr>
<td>RAM</td>
<td>0.54</td>
<td>Variables, initialized variables</td>
</tr>
</tbody>
</table>

Note: The required memory sizes vary depending on the version of the C compiler and on the compile options. The above-mentioned memory sizes include that of the memory that is used for parameter check processing (see 2.2.5).

2.1.4 File Configuration

Table 2-4 lists the files that make up this driver.

Table 2-4 VDC5 Driver File Configuration

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_vdc5.c</td>
<td>VDC5 driver API function</td>
</tr>
<tr>
<td></td>
<td>Source file defining the VDC5 driver's API functions</td>
</tr>
<tr>
<td>r_vdc5.h</td>
<td>VDC5 driver API definitions</td>
</tr>
<tr>
<td></td>
<td>Header file describing the prototypes of the VDC5 driver API functions</td>
</tr>
<tr>
<td></td>
<td>and the parameters that are defined as APIs</td>
</tr>
<tr>
<td>r_vdc5_check_parameter.c</td>
<td>VDC5 driver parameter check processing</td>
</tr>
<tr>
<td></td>
<td>Source file describing the VDC5 driver's parameter check processing</td>
</tr>
<tr>
<td>r_vdc5_check_parameter.h</td>
<td>VDC5 driver parameter check definitions</td>
</tr>
<tr>
<td></td>
<td>Header file describing the prototypes of the VDC5 driver's parameter check functions</td>
</tr>
<tr>
<td>r_vdc5_interrupt.c</td>
<td>VDC5 driver interrupt related processing</td>
</tr>
<tr>
<td></td>
<td>Source file describing the VDC5 interrupt related setup processing and</td>
</tr>
</tbody>
</table>
interrupt service routines

r_vdc5_register.c  VDC5 driver register setup processing
Source file describing the VDC5 register setup processing

r_vdc5_register.h  VDC5 driver register setup definitions
Header file describing the prototypes of the VDC5 register setup processing functions and the structures of the register address tables

r_vdc5_register_address.c  VDC5 driver register address table
File describing the table containing the VDC5's register addresses

r_vdc5_shared_param.c  VDC5 driver shared parameter processing
Source file describing the setup and retrieval processing for the parameters that are shared inside the VDC5 driver

r_vdc5_shared_param.h  VDC5 driver shared parameter definitions
Header file describing the prototypes of VDC5 driver shared parameter setup/retrieval processing functions

r_vdc5_user.h  VDC5 driver user-defined header
Header file defining compile switches and constants that can by statically edited by the user

This driver also references the following external files.

Table 2-5  External Files Referenced by the VDC5 Driver

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_typedefs.h</td>
<td>Basic type definition header</td>
</tr>
<tr>
<td></td>
<td>Header file defining the basic types</td>
</tr>
<tr>
<td>iodefine.h</td>
<td>I/O definition header</td>
</tr>
<tr>
<td></td>
<td>Header file containing the I/O definitions</td>
</tr>
</tbody>
</table>
2.1.5 **VDC5 Internal Module Configuration**

Figure 2-1 shows the internal modules of the VDC5 in the channel 0 and in a part of the channel 1 and shows data flow.

**Note:**

A. Scaler 1 receives a signal from the input controller in the other channel.
B. The output image generator block can be bypassed when it is not used.

The VDC5 is made up of 7 blocks.

1. **Input Controller:**
   Selects the input image, subjects the signals to synchronization adjustment, and adjusts the input image signals.

2. **Scaler 0/1: Graphics 0, graphics 1**
   Scaling and rotation of the input video image

3. **Image Quality Improver 0/1:**
   Image quality improvement, color conversion through color matrix function

4. **Image and VIN Synthesizer: Graphics 2, graphics 3, VIN**
   Synthesis of graphics planes and planes of video image

5. **Output Image Generator:**
   Writing and reading of image data to and from the frame buffer after the image synthesis

6. **Output Controller:**
   Output image adjustment, output format conversion, control signal output for TFT-LCD panel

7. **System Controller:**
   Interrupt control, panel clock control
2.1.6 Cascaded Connection

The VDC5 driver automatically changes the settings for cascaded connection according to its utilization conditions. When the cascaded connection settings are altered, the data flow among the VDC5's internal modules are altered (see Figure 2-2).

Cascaded connection is set to ON during the initialization performed by the VDC5 driver. In this case, scaler 0 lies below scaler 1 (see A in Figure 2-2). The data output from scaler 1 passes through the image quality improver 1 and VIN synthesizer into the upper-level image synthesizer (i.e., graphics 2).

If scaler 1 is configured to display video image or enlarge the image, the driver automatically turns off the cascaded connection (see B in Figure 2-2). When the cascaded connection is set to OFF, the data outputs from scaler 0 and scaler 1 are passed through their respective image quality improvers and synthesized together in the VIN synthesizer. The VIN synthesizer can change the way in which scaler 0 and scaler 1 are superimposed and apply image synthesis processing using alpha blending. When the use of scaler 1 in the mode such that the cascaded connection is turned off is stopped, the driver returns the setting of the cascaded connection to ON.

When the use mode such that one plane of input video image and 3 graphics planes are used is to be used, scaler 0 should be used for the input video image. In the above-mentioned case, it is not recommended to use scaler 1 for the input video image and to set the cascaded connection to OFF.

Note: A. Cascaded connection ON
B. Cascaded connection OFF

Figure 2-2 Cascaded Connection and Input Video Synthesis Processing by the VIN Synthesizer

<table>
<thead>
<tr>
<th>Input video</th>
<th>Controller</th>
<th>Scaler 0 (Graphics 0)</th>
<th>Image Quality Improver 0</th>
<th>Scaler 1 (Graphics 1)</th>
<th>Image Quality Improver 1</th>
<th>VIN Synthesizer (Graphics 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Cascaded Connection: Enabled</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input video</th>
<th>Controller</th>
<th>Scaler 0 (Graphics 0)</th>
<th>Image Quality Improver 0</th>
<th>Scaler 1 (Graphics 1)</th>
<th>Image Quality Improver 1</th>
<th>VIN Synthesizer (Graphics 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B. Cascaded Connection: Disabled</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Input video from the other channel
2.1.7 Restrictions

(1) Reserved words

The prefixes listed below are appended to the symbols such as function and variable names to be used for this driver to distinguish the driver program from other programs. Do not use in your program any names that begin with the following symbols, regardless of whether they are in upper or lower case:

- R_VDC5
- VDC5

(2) Register update

Any updates on the settings of most of the VDC5 registers are reflected on the rising edge of the vertical sync signal. Consequently, a time equivalent to up to 1 cycle of the vertical sync signal will be taken for the setting of a value to be reflected.

(3) Reentrancy

The APIs of this driver are not reentrant. The driver is likely to behave in an unexpected manner if one of its APIs is called by two or more tasks or interrupt processing routines asynchronously. Great care must be exercised with respect to the calling program of this driver and the call timing.

(4) Register accesses

This driver does not provide the user with any means of accessing all of the VDC5 registers. Some VDC5 registers are automatically set up by the driver itself.
2.2 Common Definitions
This section describes the definitions that are used commonly by the VDC5 driver routines.

2.2.1 Constant Definitions
The constant definitions are given below.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_GAM_GAIN_ADJ_NUM</td>
<td>32u</td>
<td>The number of the gamma correction gain coefficient for each signal</td>
</tr>
<tr>
<td>VDC5_GAM_START_TH_NUM</td>
<td>31u</td>
<td>The number of the gamma correction start threshold for each signal</td>
</tr>
</tbody>
</table>

2.2.2 Enumeration Type Definitions
The enumeration type definitions are given below. See 2.2.4 for the error codes.

1) vdc5_channel_t
vdc5_channel_t is an enumeration type for representing the VDC5 channels.

```c
typedef enum {
    VDC5_CHANNEL_0 = 0,
    VDC5_CHANNEL_1,
    VDC5_CHANNEL_NUM
} vdc5_channel_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_CHANNEL_0</td>
<td>0</td>
<td>Channel 0</td>
</tr>
<tr>
<td>VDC5_CHANNEL_1</td>
<td>1</td>
<td>Channel 1</td>
</tr>
<tr>
<td>VDC5_CHANNEL_NUM</td>
<td>2</td>
<td>Number of channels</td>
</tr>
</tbody>
</table>

2) vdc5_onoff_t
vdc5_onoff_t is an enumeration type for representing ON or OFF.

```c
typedef enum {
    VDC5_OFF    = 0,
    VDC5_ON     = 1
} vdc5_onoff_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_OFF</td>
<td>0</td>
<td>OFF</td>
</tr>
<tr>
<td>VDC5_ON</td>
<td>1</td>
<td>ON</td>
</tr>
</tbody>
</table>

3) vdc5_edge_t
vdc5_edge_t is an enumeration type for representing the edge of a signal.

```c
typedef enum {
    VDC5_EDGE_RISING    = 0,
    VDC5_EDGE_FALLING,  
    VDC5_EDGE_DOUBLE    
} vdc5_edge_t;
```
VDC5_EDGE_FALLING = 1
} vdc5_edge_t;

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_EDGE_RISING</td>
<td>0</td>
<td>Rising edge</td>
</tr>
<tr>
<td>VDC5_EDGE_FALLING</td>
<td>1</td>
<td>Falling edge</td>
</tr>
</tbody>
</table>

(4) vdc5_sig_pol_t

vdc5_sig_pol_t is an enumeration type for representing the polarity of a signal.

typedef enum
{
    VDC5_SIG_POL_NOT_INVERTED = 0,
    VDC5_SIG_POL_INVERTED     = 1
} vdc5_sig_pol_t;

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_SIG_POL_NOT_INVERTED</td>
<td>0</td>
<td>Not inverted</td>
</tr>
<tr>
<td>VDC5_SIG_POL_INVERTED</td>
<td>1</td>
<td>Inverted</td>
</tr>
</tbody>
</table>

(5) vdc5_scaling_type_t

vdc5_scaling_type_t is an enumeration type for representing the types of scalers.

typedef enum
{
    VDC5_SC_TYPE_SC0 = 0,
    VDC5_SC_TYPE_SC1,
    VDC5_SC_TYPE_OIR,
    VDC5_SC_TYPE_NUM
} vdc5_scaling_type_t;

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_SC_TYPE_SC0</td>
<td>0</td>
<td>Scaler 0</td>
</tr>
<tr>
<td>VDC5_SC_TYPE_SC1</td>
<td>1</td>
<td>Scaler 1</td>
</tr>
<tr>
<td>VDC5_SC_TYPE_OIR</td>
<td>2</td>
<td>Output image generator</td>
</tr>
<tr>
<td>VDC5_SC_TYPE_NUM</td>
<td>3</td>
<td>Number of scaler types</td>
</tr>
</tbody>
</table>

(6) vdc5_graphics_type_t

vdc5_graphics_type_t is an enumeration type for representing the types of graphics.

typedef enum
{
    VDC5_GR_TYPE_GR0 = 0,
    VDC5_GR_TYPE_GR1,
    VDC5_GR_TYPE_GR2,
    VDC5_GR_TYPE_GR3,
    VDC5_GR_TYPE_VIN,
    VDC5_GR_TYPE_OIR,
    VDC5_GR_TYPE_NUM
} vdc5_graphics_type_t;
The VDC5 has four layers (graphics 0 (layer 0) to graphics 3 (layer 3)) and has an output image generator layer. Of the VDC5’s internal blocks, scaler 0 and scaler 1 correspond to graphics 0 and graphics 1, respectively, and the two image synthesizer blocks correspond to graphics 2 and graphics 3, respectively. The VIN synthesizer layer is not a substantial layer.

Each of the scalers and output image generator can be divided into the former stage for writing the input data and the latter stage for reading data from memory (see A in Figure 2-3). The former stage of the scalers performs scale-down and rotation processing on the input image data and writes the results into memory. The latter stage of the scalers performs scale-up processing on the data that is read from memory. Scaler 1 can blend the data read from the memory in the latter stage with the image data from the lower-layer (i.e., scaler 0). The output image generator can perform none of scale-down, rotation, and scale-up processing. Different layer IDs, which are defined in the enumeration type vdc5_layer_id_t, are assigned to the memory write processing and read processing for the same layer.
The image data read from memory and the image data from the lower layer can be blended in the image synthesizer (see B in Figure 2-3).

### Figure 2-3 Memory Write/Read Processing

When the cascaded connection is set to ON, the results of image synthesis among layers look like as shown in Figure 2-4. Layer 0 is the bottom layer and layer 3 is the top layer.

### Figure 2-4 Layers and Image Synthesis
vdc5_int_type_t is an enumeration type for representing the types of VDC5 interrupts.

```c
typedef enum {
    VDC5_INT_TYPE_S0_VI_VSYNC = 0,
    VDC5_INT_TYPE_S0_LO_VSYNC,
    VDC5_INT_TYPE_S0_VSYNCERR,
    VDC5_INT_TYPE_VLINE,
    VDC5_INT_TYPE_IV1_VBUFERR,
    VDC5_INT_TYPE_IV3_VBUFERR,
    VDC5_INT_TYPE_IV5_VBUFERR,
    VDC5_INT_TYPE_IV6_VBUFERR,
    VDC5_INT_TYPE_S0_WLINE,
    VDC5_INT_TYPE_S1_VI_VSYNC,
    VDC5_INT_TYPE_S1_LO_VSYNC,
    VDC5_INT_TYPE_S1_VSYNCERR,
    VDC5_INT_TYPE_S1_VFIELD,
    VDC5_INT_TYPE_IV2_VBUFERR,
    VDC5_INT_TYPE_IV4_VBUFERR,
    VDC5_INT_TYPE_S1_WLINE,
    VDC5_INT_TYPE_OIR_VI_VSYNC,
    VDC5_INT_TYPE_OIR_LO_VSYNC,
    VDC5_INT_TYPE_OIR_VLINE,
    VDC5_INT_TYPE_OIR_VFIELD,
    VDC5_INT_TYPE_OIR_VBUFERR,
    VDC5_INT_TYPE_IV7_VBUFERR,
    VDC5_INT_TYPE_IV8_VBUFERR,
    VDC5_INT_TYPE_NUM
} vdc5_int_type_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_INT_TYPE_S0_VI_VSYNC</td>
<td>0</td>
<td>Vsync signal input to scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S0_LO_VSYNC</td>
<td>1</td>
<td>Vsync signal output from scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S0_VSYNCERR</td>
<td>2</td>
<td>Missing Vsync signal for scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_VLINE</td>
<td>3</td>
<td>Specified line signal for panel output in graphics 3</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S0_VFIELD</td>
<td>4</td>
<td>Field end signal for recording function in scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV1_VBUFERR</td>
<td>5</td>
<td>Frame buffer write overflow signal for scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV3_VBUFERR</td>
<td>6</td>
<td>Frame buffer read underflow signal for graphics 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV5_VBUFERR</td>
<td>7</td>
<td>Frame buffer read underflow signal for graphics 2</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV6_VBUFERR</td>
<td>8</td>
<td>Frame buffer read underflow signal for graphics 3</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S0_WLINE</td>
<td>9</td>
<td>Write specification line signal input to scaling-down control block in scaler 0</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S1_VI_VSYNC</td>
<td>10</td>
<td>Vsync signal input to scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S1_LO_VSYNC</td>
<td>11</td>
<td>Vsync signal output from scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S1_VSYNCERR</td>
<td>12</td>
<td>Missing Vsync signal for scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S1_VFIELD</td>
<td>13</td>
<td>Field end signal for recording function in scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV2_VBUFERR</td>
<td>14</td>
<td>Frame buffer write overflow signal for scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_IV4_VBUFERR</td>
<td>15</td>
<td>Frame buffer read underflow signal for graphics 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_S1_WLINE</td>
<td>16</td>
<td>Write specification line signal input to scaling-down control block in scaler 1</td>
</tr>
<tr>
<td>VDC5_INT_TYPE_OIR_VI_VSYNC</td>
<td>17</td>
<td>Vsync signal input to output image generator</td>
</tr>
</tbody>
</table>
### VDC5_INT_TYPE_OIR_LO_VSYNC

Vsync signal output from output image generator

### VDC5_INT_TYPE_OIR_VLINE

Specified line signal for panel output in output image generator

### VDC5_INT_TYPE_OIR_VFIELD

Field end signal for recording function in output image generator

### VDC5_INT_TYPE_IV7_VBUFERR

Frame buffer write overflow signal for output image generator

### VDC5_INT_TYPE_IV8_VBUFERR

Frame buffer read underflow signal for graphics (OIR)

### VDC5_INT_TYPE_NUM

Number of VDC5 interrupt types

---

#### vdc5_gr_disp_sel_t

vdc5_gr_disp_sel_t is an enumeration type for representing the graphics display modes.

```c
typedef enum
{
    VDC5_DISPSEL_IGNORED    = -1,
    VDC5_DISPSEL_BACK       = 0,
    VDC5_DISPSEL_LOWER      = 1,
    VDC5_DISPSEL_CURRENT    = 2,
    VDC5_DISPSEL_BLEND      = 3,
    VDC5_DISPSEL_NUM        = 4
} vdc5_gr_disp_sel_t;
```

#### Enumeration constant

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ignored, no change made</td>
<td>-1</td>
</tr>
<tr>
<td>Background color display</td>
<td>0</td>
</tr>
<tr>
<td>Lower-layer graphics display</td>
<td>1</td>
</tr>
<tr>
<td>Current graphics display</td>
<td>2</td>
</tr>
<tr>
<td>Blended display of lower-layer graphics and current graphics</td>
<td>3</td>
</tr>
<tr>
<td>Number of graphics display modes</td>
<td>4</td>
</tr>
</tbody>
</table>

#### vdc5_imgimprv_id_t

vdc5_imgimprv_id_t is an enumeration type for representing the image quality improvers.

```c
typedef enum
{
    VDC5_IMG_IMPRV_0 = 0,
    VDC5_IMG_IMPRV_1,
    VDC5_IMG_IMPRV_NUM
} vdc5_imgimprv_id_t;
```

#### Enumeration constant

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image quality improver 0</td>
<td>0</td>
</tr>
<tr>
<td>Image quality improver 1</td>
<td>1</td>
</tr>
<tr>
<td>Number of image quality improvers</td>
<td>2</td>
</tr>
</tbody>
</table>

### 2.2.3 Structure Definitions

#### vdc5_period_rect_t

vdc5_period_rect_t is a structure for representing the horizontal/vertical timing of the VDC5 signals.
typedef struct {
    uint16_t    vs;
    uint16_t    vw;
    uint16_t    hs;
    uint16_t    hw;
} vdc5_period_rect_t;

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>vs</td>
<td>Vertical signal start position from the reference signal (lines)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>vw</td>
<td>Vertical signal width (lines)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>hs</td>
<td>Horizontal signal start position from the reference signal (clock cycles)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>hw</td>
<td>Horizontal signal width (clock cycles)</td>
</tr>
</tbody>
</table>

The horizontal/vertical timings in the `vdc5_period_rect_t` structure are represented as a rectangle area as shown in Figure 2-5.

![Figure 2-5 Rectangle Representing the Horizontal and Vertical Timings](image_url)
### 2.2.4 Error Codes

Table 2-6 shows a list of error codes of the VDC5 driver.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Value</th>
<th>Description (Error Type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_OK</td>
<td>0</td>
<td>Normal termination</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_CHANNEL</td>
<td>1</td>
<td>Invalid channel error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An illegal channel is specified.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_LAYER_ID</td>
<td>2</td>
<td>Invalid layer ID error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An illegal layer ID is specified.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_NULL</td>
<td>3</td>
<td>NULL specification error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NULL is specified for a required parameter.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_BIT_WIDTH</td>
<td>4</td>
<td>Bit width error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A value exceeding the possible bit width is specified.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_UNDEFINED</td>
<td>5</td>
<td>Undefined parameter error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A value that is not defined in the specification is specified.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_EXCEED_RANGE</td>
<td>6</td>
<td>Out-of-value-range error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The specified parameter value is beyond the value range defined in the specification.</td>
</tr>
<tr>
<td>VDC5_ERR_PARAM_CONDITION</td>
<td>7</td>
<td>Unauthorized condition error (parameter error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A parameter is specified under conditions that are not authorized by the specification.</td>
</tr>
<tr>
<td>VDC5_ERR_IF_CONDITION</td>
<td>8</td>
<td>Interface condition error (interface error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An API function is called under unauthorized conditions.</td>
</tr>
<tr>
<td>VDC5_ERR_RESOURCE_CLK</td>
<td>9</td>
<td>Clock resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No panel clock is set up.</td>
</tr>
<tr>
<td>VDC5_ERR_RESOURCE_VSYNC</td>
<td>10</td>
<td>Vertical sync signal resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No vertical sync signal is set up.</td>
</tr>
<tr>
<td>VDC5_ERR_RESOURCE_INPUT</td>
<td>11</td>
<td>Input signal resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No video image input is set up.</td>
</tr>
<tr>
<td>VDC5_ERR_RESOURCE_OUTPUT</td>
<td>12</td>
<td>Output resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No display output is set up.</td>
</tr>
<tr>
<td>VDC5_ERRRESOURCE_LVDS_CLK</td>
<td>13</td>
<td>LVDS clock resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The LVDS clock is not set up when it is specified to use, or the LVDS clock is already set up when it is specified to set up.</td>
</tr>
<tr>
<td>VDC5_ERRRESOURCE_LAYER</td>
<td>14</td>
<td>Layer resource error (resource error)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The specified layer is under unavailable conditions.</td>
</tr>
</tbody>
</table>
2.2.5 Compile Switch

The following compile switch is defined in "r_vdc5_user.h" for this driver.

<table>
<thead>
<tr>
<th>Compile Switch</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_VDC5_CHECK_PARAMETERS</td>
<td>Enabling this definition causes the parameter check of the VDC5 driver API functions when they are called. If an error is found as the result of the parameter check, an error code indicating a parameter error is returned. See 2.2.4 for the error codes.</td>
</tr>
</tbody>
</table>
2.2.6 User Custom Parameters

Parameters that can statically be changed by the user are defined in "r_vdc5_user.h" for this driver.

(1) Enumeration type vdc5_colcnv_rgb_ycbcr_t

vdc5_colcnv_rgb_ycbcr_t is an enumeration type for representing the color matrix values. It is referenced by the VDC5 driver when converting GBR signals to YCbCr signals. The default values are the standard values that are described in the hardware manual.

```c
typedef enum {
    VDC5_COLORCONV_Y_R  = (77u),
    VDC5_COLORCONV_Y_G  = (150u),
    VDC5_COLORCONV_Y_B  = (29u),
    VDC5_COLORCONV_CB_R = (2005u),
    VDC5_COLORCONV_CB_G = (1963u),
    VDC5_COLORCONV_CB_B = (128u),
    VDC5_COLORCONV_CR_R = (128u),
    VDC5_COLORCONV_CR_G = (1941u),
    VDC5_COLORCONV_CR_B = (2027u)
} vdc5_colcnv_rgb_ycbcr_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_COLORCONV_Y_R</td>
<td>77u</td>
<td>Cr/R Signal Gain Adjustment for Y/G Signal Output (0.299)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_Y_G</td>
<td>150u</td>
<td>Y/G Signal Gain Adjustment for Y/G Signal Output (0.587)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_Y_B</td>
<td>29u</td>
<td>Cb/B Signal Gain Adjustment for Y/G Signal Output (0.114)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CB_R</td>
<td>2005u</td>
<td>Cr/R Signal Gain Adjustment for Cb/B Signal Output (-0.169)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CB_G</td>
<td>1963u</td>
<td>Y/G Signal Gain Adjustment for Cb/B Signal Output (-0.331)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CB_B</td>
<td>128u</td>
<td>Cb/B Signal Gain Adjustment for Cb/B Signal Output (0.500)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CR_R</td>
<td>128u</td>
<td>Cr/R Signal Gain Adjustment for Cr/R Signal Output (0.500)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CR_G</td>
<td>1941u</td>
<td>Y/G Signal Gain Adjustment for Cr/R Signal Output (-0.419)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_CR_B</td>
<td>2027u</td>
<td>Cb/B Signal Gain Adjustment for Cr/R Signal Output (-0.081)</td>
</tr>
</tbody>
</table>

**Note:** The values are represented by 2's complement of the 11-bit values. \((-1024 \sim +1023[LSB] , \ 256[LSB] = 1.0[time])\)

(2) Enumeration type vdc5_colcnv_ycbcr_rgb_t

vdc5_colcnv_ycbcr_rgb_t is an enumeration type for representing the color matrix values. It is referenced by the VDC5 driver when converting YCbCr signals to GBR signals. The default values are the standard values that are described in the hardware manual.

```c
typedef enum {
    VDC5_COLORCONV_G_Y  = (256u),
    VDC5_COLORCONV_G_CB = (1960u),
} vdc5_colcnv_ycbcr_rgb_t;
```
VDC5_COLORCONV_G_CR = (1865u),  
VDC5_COLORCONV_B_Y = (256u),  
VDC5_COLORCONV_B_CB = (454u),  
VDC5_COLORCONV_B_CR = (0u),  
VDC5_COLORCONV_R_Y = (256u),  
VDC5_COLORCONV_R_CB = (0u),  
VDC5_COLORCONV_R_CR = (359u)

) vdc5_colconv_ycbcr_rgb_t;

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_COLORCONV_G_Y</td>
<td>256u</td>
<td>Cr/R Signal Gain Adjustment for Y/G Signal Output (1.000)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_G_CB</td>
<td>1960u</td>
<td>Y/G Signal Gain Adjustment for Y/G Signal Output (-0.344)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_G_CR</td>
<td>1865u</td>
<td>Cb/B Signal Gain Adjustment for Y/G Signal Output (-0.714)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_B_Y</td>
<td>256u</td>
<td>Cr/R Signal Gain Adjustment for Cb/B Signal Output (1.000)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_B_CB</td>
<td>454u</td>
<td>Y/G Signal Gain Adjustment for Cb/B Signal Output (1.772)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_B_CR</td>
<td>0u</td>
<td>Cb/B Signal Gain Adjustment for Cb/B Signal Output (0.000)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_R_Y</td>
<td>256u</td>
<td>Cr/R Signal Gain Adjustment for Cr/R Signal Output (1.000)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_R_CB</td>
<td>0u</td>
<td>Y/G Signal Gain Adjustment for Cr/R Signal Output (0.000)</td>
</tr>
<tr>
<td>VDC5_COLORCONV_R_CR</td>
<td>359u</td>
<td>Cb/B Signal Gain Adjustment for Cr/R Signal Output (1.402)</td>
</tr>
</tbody>
</table>

Note: The values are represented by 2's complement of the 11-bit values.  
(-1024 ~ +1023[LSB], 256[LSB] = 1.0[time])

(3) Constant definitions
The constants are described below.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_COLORCONV_DC_OFFSET</td>
<td>128u</td>
<td>Offset (DC) adjustment values for the Y/G, B, R signals in a color matrix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 [-128] to 255 [+127], 128[LSB] = 0) Referenced by the VDC5 driver when setting up the color matrix.</td>
</tr>
<tr>
<td>VDC5_COLORCONV_1TIMES_GAIN</td>
<td>256u</td>
<td>1.0[time] gain value for the color matrix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1024 to +1023[LSB], 256[LSB] = 1.0[time] Referenced by the VDC5 driver when converting YCbCr signals to YCbCr signals and when converting GBR signals to GBR signals.</td>
</tr>
<tr>
<td>VDC5_LVDS_PLL_WAIT_CYCLE</td>
<td>38u</td>
<td>Number of cycles through the loop for generating 1 usec of busy wait period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced when setting up the LVDS PLL.</td>
</tr>
<tr>
<td>VDC5_LVDS_PLL_WAIT_200USEC</td>
<td>13400u</td>
<td>Number of cycles through the loop for generating 200 usec of busy wait period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced when setting up the LVDS PLL.</td>
</tr>
</tbody>
</table>
2.3 API Functions

The API functions of the VDC5 driver are listed in Table 2-8.

<table>
<thead>
<tr>
<th>Function name</th>
<th>Section</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_VDC5_Initialize</td>
<td>2.3.1</td>
<td>VDC5 driver initialization</td>
</tr>
<tr>
<td>R_VDC5_Terminate</td>
<td>2.3.2</td>
<td>VDC5 driver termination</td>
</tr>
<tr>
<td>R_VDC5_VideoInput</td>
<td>2.3.3</td>
<td>Video input setup</td>
</tr>
<tr>
<td>R_VDC5_SyncControl</td>
<td>2.3.4</td>
<td>Synchronization control setup</td>
</tr>
<tr>
<td>R_VDC5_DisplayOutput</td>
<td>2.3.5</td>
<td>Display output setup</td>
</tr>
<tr>
<td>R_VDC5_CallbackISR</td>
<td>2.3.6</td>
<td>Interrupt callback setup</td>
</tr>
<tr>
<td>R_VDC5_WriteDataControl</td>
<td>2.3.7</td>
<td>Data write control processing</td>
</tr>
<tr>
<td>R_VDC5_ChangeWriteProcess</td>
<td>2.3.8</td>
<td>Data write change processing</td>
</tr>
<tr>
<td>R_VDC5_ReadDataControl</td>
<td>2.3.9</td>
<td>Data read control processing</td>
</tr>
<tr>
<td>R_VDC5_ChangeReadProcess</td>
<td>2.3.10</td>
<td>Data read change processing</td>
</tr>
<tr>
<td>R_VDC5_StartProcess</td>
<td>2.3.11</td>
<td>Data write/read start processing</td>
</tr>
<tr>
<td>R_VDC5_StopProcess</td>
<td>2.3.12</td>
<td>Data write/read stop processing</td>
</tr>
<tr>
<td>R_VDC5_ReleaseDataControl</td>
<td>2.3.13</td>
<td>Data write/read control release process</td>
</tr>
<tr>
<td>R_VDC5_VideoNoiseReduction</td>
<td>2.3.14</td>
<td>Noise reduction setup</td>
</tr>
<tr>
<td>R_VDC5_ImageColorMatrix</td>
<td>2.3.15</td>
<td>Color matrix setup</td>
</tr>
<tr>
<td>R_VDC5_ImageEnhancement</td>
<td>2.3.16</td>
<td>Image enhancement processing</td>
</tr>
<tr>
<td>R_VDC5_ImageBlackStretch</td>
<td>2.3.17</td>
<td>Black stretch setup</td>
</tr>
<tr>
<td>R_VDC5_AlphaBlending</td>
<td>2.3.18</td>
<td>Alpha blending setup</td>
</tr>
<tr>
<td>R_VDC5_AlphaBlendingRect</td>
<td>2.3.19</td>
<td>Rectangle alpha blending setup</td>
</tr>
<tr>
<td>R_VDC5_Chromakey</td>
<td>2.3.20</td>
<td>Chroma key setup</td>
</tr>
<tr>
<td>R_VDC5_CLUT</td>
<td>2.3.21</td>
<td>CLUT setup</td>
</tr>
<tr>
<td>R_VDC5_DisplayCalibration</td>
<td>2.3.22</td>
<td>Display calibration processing</td>
</tr>
<tr>
<td>R_VDC5_GammaCorrection</td>
<td>2.3.23</td>
<td>Gamma correction setup</td>
</tr>
<tr>
<td>R_VDC5_GetISR</td>
<td>2.3.24</td>
<td>Interrupt service routine acquisition processing</td>
</tr>
</tbody>
</table>
2.3.1 R_VDC5_Initialize

Synopsis  
VDC5 driver initialization

Header  
r_vdc5.h

Declaration  
vdc5_error_t R_VDC5_Initialize(  
    const vdc5_channel_t ch,  
    const vdc5_init_t * const param,  
    void (* const init_func)(uint32_t),  
    const uint32_t user_num);

Arguments  
- vdc5_channel_t ch: Channel  
  — VDC5_CHANNEL_0: Channel 0  
  — VDC5_CHANNEL_1: Channel 1  
- vdc5_init_t * param: Initialization parameter  
- void (* init_func)(uint32_t): Pointer to a user-defined function  
  Specify the user-implemented function that is to be executed together with the VDC5  
  driver initialization processing. Within the API function R_VDC5_Initialize, this function  
  is called before the VDC5 registers are set up. user_num is used as the argument  
  when the function is called. Specify '0' when this function is not required.  
- uint32_t user_num: User defined number  
  Specify the argument to be passed to the user-defined function init_func. This  
  parameter is ignored if '0' is specified as the user-defined function.

Return value  
- vdc5_error_t: Error code  
  — VDC5_OK: Normal termination  
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error  
  — VDC5_ERR_PARAM_NULL: NULL specification error  
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error  
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error  
  — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error  
  — VDC5_ERR_PARAM_CONDITION: Unauthorized condition error  
  — VDC5_ERR_RESOURCE_LVDS_CLK: LVDS clock resource error

Details  
(1) Function  
This function initializes the VDC5 driver and performs the following associated processing:
- Initializes the VDC5 driver's internal variables.  
- Calls the user-defined function specified in init_func.  
- Sets up and enables the VDC5's panel clock.  
- Sets up and enables the LVDS and LVDS PLL.  
- Disables all the VDC5 interrupts.

When the LVDS PLL is set in this function, it is required to wait for 200 usec. In this driver, the 200-usec waiting  
process is implemented in the function Wait_200_usec (source file "r_vdc5_register.c") using busy wait. Change the  
function, if necessary.

(2) Use conditions
The following steps of processing need to be performed before the VDC5 driver is started:

- Supply of a clock to the VDC5 modules
- Setup of VDC5-related interrupts (interrupt service routine, interrupt priority)
- Setup of VDC5-related I/O ports
- Environment-specific setup necessary for the LCD panel and video input

Execute the above-listed steps except for setup of I/O ports for the LCD panel output before calling this function or implement a function that performs the above-listed steps and specify it in init_func as a user-defined function.

The function returns the LVDS clock resource error (VDC5_ERRRESOURCE_LVDS_CLK) if an attempt is made to use the LVDS clock without setting it up or to override the existing LVDS setup.

(3) Parameter details

The members of the vdc5_init_t structure are described below.

```c
typedef struct
{
    vdc5_panel_clksel_t    panel_icksel;
    vdc5_panel_clk_dcdr_t  panel_dcdr;
    const vdc5_lvds_t     * lvds;
} vdc5_init_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_panel_clksel_t</td>
<td>panel_icksel</td>
<td>Panel clock select</td>
</tr>
<tr>
<td>vdc5_panel_clksel_t</td>
<td>VDC5_PANEL_ICKSEL_IMG:</td>
<td>Frequency-divided video image clock (VIDEO_X1)</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_IMG_DV:</td>
<td>Frequency-divided video image clock (DV_CLK)</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_EXT_0:</td>
<td>Frequency-divided external clock 0 (LCD0_EXTCLK)</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_EXT_1:</td>
<td>Frequency-divided external clock 1 (LCD1_EXTCLK)</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_PERI:</td>
<td>Frequency-divided peripheral clock 1 (P1φ)</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_LVDS:</td>
<td>LVDS PLL clock</td>
</tr>
<tr>
<td></td>
<td>VDC5_PANEL_ICKSEL_LVDS_DIV7:</td>
<td>LVDS PLL clock divided by 7</td>
</tr>
</tbody>
</table>

| vdc5_panel_clk_dcdr_t | panel_dcdr                | Clock frequency division ratio                   |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_1:    | 1/1                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_2:    | 1/2                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_3:    | 1/3                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_4:    | 1/4                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_5:    | 1/5                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_6:    | 1/6                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_7:    | 1/7                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_8:    | 1/8                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_9:    | 1/9                                              |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_12:   | 1/12                                             |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_16:   | 1/16                                             |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_24:   | 1/24                                             |
| vdc5_panel_clk_dcdr_t | VDC5_PANEL_CLKDIV_1_32:   | 1/32                                             |

This parameter is not referenced when panel_icksel is set to VDC5_PANEL_ICKSEL_LVDS or
When a video clock (DC5_PANEL_ICKSEL_IMG or VDC5_PANEL_ICKSEL_IMG_DV) is selected in the panel clock select member (panel_icksel), an unauthorized condition error (VDC5_ERR_PARAM_CONDITION) is returned if the setup of the video input has already been made and this selection conflicts with the input selection (inp_sel).

The members of the vdc5_lvds_t structure are described below.

```c
typedef struct {
    vdc5_lvds_in_clk_sel_t  lvds_in_clk_sel;
    vdc5_lvds_ndiv_t        lvds_idiv_set;
    uint16_t                lvdspll_tst;
    vdc5_lvds_ndiv_t        lvds_odiv_set;
    vdc5_channel_t          lvds_vdc_sel;
    uint16_t                lvdspll_fd;
    uint16_t                lvdspll_rd;
    vdc5_lvds_pll_nod_t     lvdspll_od;
} vdc5_lvds_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_lvds_in_clk_sel_t</td>
<td>lvds_in_clk_sel</td>
<td>Clock input to frequency divider 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_IMG: VIDEO_X1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_DV_0: DV0_CLK0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_DV_1: DV1_CLK1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_EXT_0: LCD0_EXTCLK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_EXT_1: LCD1_EXTCLK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_INCLK_SEL_PERI: P1φ</td>
</tr>
<tr>
<td>vdc5_lvds_ndiv_t</td>
<td>lvds_idiv_set</td>
<td>Frequency dividing value (NIDIV) for frequency divider 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_1: NIDIV = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_2: NIDIV = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_4: NIDIV = 4</td>
</tr>
<tr>
<td>uint16_t</td>
<td>lvdspll_tst</td>
<td>Internal parameter setting for LVDS PLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter should be 16.</td>
</tr>
<tr>
<td>vdc5_lvds_ndiv_t</td>
<td>lvds_odiv_set</td>
<td>Frequency dividing value (NODIV) for frequency divider 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_1: NODIV = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_2: NODIV = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_LVDS_NDIV_4: NODIV = 4</td>
</tr>
<tr>
<td>vdc5_channel_t</td>
<td>lvds_vdc_sel</td>
<td>Channel select in video display controller 5 whose data is to be output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>through</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_CHANNEL_0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_CHANNEL_1</td>
</tr>
<tr>
<td>uint16_t</td>
<td>lvdspll_fd</td>
<td>Frequency dividing value (NFD) for the feedback frequency in the LVDS PLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NFD = lvdspll_fd (24 ~ 2047)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The following values are not allowed: 28 to 31, 37 to 39, 46, 47, and 55</td>
</tr>
<tr>
<td>uint16_t</td>
<td>lvdspll_rd</td>
<td>Frequency dividing value (N RD) for the input frequency in the LVDS PLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NRD = lvdspll_rd + 1</td>
</tr>
</tbody>
</table>

Specify NULL if this parameter is not required.
When LVDS PLL divide-by-7 clock (VDC5_PANEL_ICKSEL_LVDS_DIV7) is selected in the panel clock select member (pnael_icksel), an unauthorized condition error (VDC5_ERR_PARAM_CONDITION) is returned if the VDC5 channel select (lvds_vdc_sel) which is output from the LVDS specified in the LVDS-related parameter differs from the specified channel (ch).
2.3.2  R_VDC5_Terminate

Synopsis
VDC5 driver termination

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_Terminate(
    const vdc5_channel_t  ch,
    void    (* const quit_func)(uint32_t),
    const uint32_t   user_num);

Arguments
•  vdc5_channel_t ch: Channel
  —  VDC5_CHANNEL_0: Channel 0
  —  VDC5_CHANNEL_1: Channel 1
•  void (* quit_func)(uint32_t): Pointer to a user-defined function
  Specify the user-implemented function that is to be executed together with the VDC5
  driver termination processing. Within the API function R_VDC5_Terminate, this
  function is called after the VDC5 registers are set up. user-num is used as the
  argument when the function is called. Specify '0' when this function is not required.
•  uint32_t user_num: User defined number
  Specify the argument to be passed to the user-defined function quit_func. This
  parameter is ignored if '0' is specified as the user-defined function.

Return value
•  vdc5_error_t: Error code
  —  VDC5_OK: Normal termination
  —  VDC5_ERR_PARAM_CHANNEL: Channel invalid error

Details
(1)  Function
This function terminates the VDC5 driver and performs the following associated processing:
•  Disables all the VDC5 interrupts.
•  Disables the VDC5 panel clock.
•  Disables the LVDS if one is used and becomes unnecessary as the result of calling this function.
•  Calls the user-defined function specified in quit_func.

(2)  Use conditions
There are no particular conditions with respect to the call of this function.
2.3.3 R_VDC5_VideoInput

Synopsis
Video input setup

Header
r_vdc5.h

Declaration
dvc5_error_t R_VDC5_VideoInput(
    const vdc5_channel_t ch,
    const vdc5_input_t * const param);

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_input_t * param: Video input setup parameter
  Do not specify NULL.

Return value
• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_NULL: NULL specification error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  — VDC5_ERR_PARAM_CONDITION: Unauthorized condition error

Details

(1) Function
This function performs the following processing on the video input:
• Selects the video input.
• Sets up the phase timing of the input signals.
• Performs delay control on the sync signal for the video inputs.
• Sets up the parameters for the external input video signals only when they are used.

(2) Use conditions
There are no particular conditions with respect to the call of this function.

(3) Parameter details
The members of the vdc5_input_t structure are described below.

typedef struct
defined
{
    vdc5_input_sel_t            inp_sel;
    uint16_t                    inp_fh50;
    uint16_t                    inp_fh25;
    const vdc5_sync_delay_t * dly;
    const vdc5_ext_in_sig_t * ext_sig;
} vdc5_input_t;
The input select setting (inp_sel) must not conflict with the VDC5 panel clock select setting. When this function is called and the VDC5 panel clock for the same channel has been set up, only the input select settings listed below are valid. Any other settings would cause the driver to return an unauthorized condition error (VDC5_ERR_PARAM_CONDITION).

- When the panel clock select is set to a frequency-divided video clock (VIDEO_X1):
  Set the input select to the video decoder output signals (VDC5_INPUT_SEL_VDEC).
- When the panel clock select is set to a frequency-divided video clock (DV_CLK):
  Set the input select to the signals supplied via the external input pins (VDC5_INPUT_SEL_EXT).
- When the panel clock select is set to a value other than the video clocks:
  The input select may be set to either valid value.

The members of the vdc5_sync_delay_t structure are described below.

```
typedef struct
{
  uint16_t   inp_vs_dly_l;
  uint16_t   inp_fld_dly;
  uint16_t   inp_vs_dly;
  uint16_t   inp_hs_dly;
} vdc5_sync_delay_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>inp_vs_dly_l</td>
<td>0</td>
<td>Number of lines for delaying vsync signal and field differentiation signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 to 7 [lines]</td>
</tr>
<tr>
<td>uint16_t</td>
<td>inp_fld_dly</td>
<td>0</td>
<td>Field differentiation signal delay amount</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 to 254 [clock cycles]</td>
</tr>
</tbody>
</table>
The members of the `vdc5_ext_in_sig_t` structure are described below.

```c
typedef struct
{
    vdc5_extin_format_t     inp_format;
    vdc5_edge_t             inp_pxd_edge;
    vdc5_edge_t             inp_vs_edge;
    vdc5_edge_t             inp_hs_edge;
    vdc5_onoff_t            inp_endian_on;
    vdc5_onoff_t            inp_swap_on;
    vdc5_sig_pol_t          inp_vs_inv;
    vdc5_sig_pol_t          inp_hs_inv;
    vdc5_extin_ref_hsync_t  inp_h_edge_sel;
    vdc5_extin_input_line_t inp_f525_625;
    vdc5_extin_h_pos_t      inp_h_pos;
} vdc5_ext_in_sig_t;
```

### Type | Member Name | Description
---|---|---
| vdc5_extin_format_t | inp_format | External input format select
| | | - VDC5_EXTIN_FORMAT_RGB888 (0): RGB888
| | | - VDC5_EXTIN_FORMAT_RGB666 (1): RGB666
| | | - VDC5_EXTIN_FORMAT_RGB565 (2): RGB565
| | | - VDC5_EXTIN_FORMAT_BT656 (3): BT6556
| | | - VDC5_EXTIN_FORMAT_BT601 (4): BT6501
| | | - VDC5_EXTIN_FORMAT_YCBCR422 (5): YCbCr422
| | | - VDC5_EXTIN_FORMAT_YCBCR444 (6): YCbCr444
| vdc5_edge_t | inp_pxd_edge | Clock edge select for capturing external input video image signals DV_DATA23 to DV_DATA0
| | | - VDC5_EDGE_RISING: Rising edge
| | | - VDC5_EDGE_FALLING: Falling edge
| vdc5_edge_t | inp_vs_edge | Clock edge select for capturing external input Vsync signals DV_VSYNC
| | | - VDC5_EDGE_RISING: Rising edge
| | | - VDC5_EDGE_FALLING: Falling edge
| vdc5_edge_t | inp_hs_edge | Clock edge select for capturing external input Hsync signals DV_HSYNC
| | | - VDC5_EDGE_RISING: Rising edge
| | | - VDC5_EDGE_FALLING: Falling edge
| vdc5_onoff_t | inp_endian_on | External input bit endian change on/off control
| | | - VDC5_OFF
| | | - VDC5_ON
| vdc5_onoff_t | inp_swap_on | External input B/R signal swap on/off control
| | | - VDC5_OFF
| | | - VDC5_ON
| vdc5_sig_pol_t | inp_vs_inv | External input Vsync signal DV_VSYNC inversion control
| | | - VDC5_SIG_POL_NOT_INVERTED: Not inverted (positive polarity)
| | | - VDC5_SIG_POL_INVERTED: Inverted (negative polarity)
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_sig_pol_t</td>
<td>External input Hsync signal DV_HSYNC inversion control</td>
</tr>
<tr>
<td>inp_hs_inv</td>
<td>• VDC5_SIG_POL_NOT_INVERTED: Not inverted (positive polarity)</td>
</tr>
<tr>
<td></td>
<td>• VDC5_SIG_POL_INVERTED: Inverted (negative polarity)</td>
</tr>
<tr>
<td>vdc5_extin_ref_hsync_t</td>
<td>Reference select for external input BT656 Hsync signal</td>
</tr>
<tr>
<td>inp_h_edge_sel</td>
<td>• VDC5_EXTIN_REF_H_EAV (0): EAV</td>
</tr>
<tr>
<td></td>
<td>• VDC5_EXTIN_REF_H_SAV (1): SAV</td>
</tr>
<tr>
<td>vdc5_extin_input_line_t</td>
<td>Number of lines for BT656 external input</td>
</tr>
<tr>
<td>inp_f525_625</td>
<td>• VDC5_EXTIN_LINE_525 (0): 525 lines</td>
</tr>
<tr>
<td></td>
<td>• VDC5_EXTIN_LINE_625 (1): 625 lines</td>
</tr>
<tr>
<td>vdc5_extin_h_pos_t</td>
<td>Y/Cb/Y/Cr data string start timing to Hsync reference</td>
</tr>
<tr>
<td>inp_h_pos</td>
<td>• VDC5_EXTIN_H_POS_CBYCRY (0): Cb/Y/Cr/Y (BT656/601), Cb/Cr (YCbCr422)</td>
</tr>
<tr>
<td></td>
<td>• VDC5_EXTIN_H_POS_YCRYCB (1): Y/Cr/Y/Cb (BT656/601), inhibited (YCbCr422)</td>
</tr>
<tr>
<td></td>
<td>• VDC5_EXTIN_H_POS_CRYCBY (2): Cb/Cr/Y/Cb (BT656/601), inhibited (YCbCr422)</td>
</tr>
<tr>
<td></td>
<td>• VDC5_EXTIN_H_POS_YCBYCR (3): Y/Cb/Cr/Y/Cr (BT656/601), Cb/Cr (YCbCr422)</td>
</tr>
</tbody>
</table>

The function returns an unauthorized condition error (VDC5_ERR_PARAM_CONDITION) if the data string start timing to Hsync reference (inp_h_pos) is set to VDC5_EXTIN_H_POS_YCRYCB or VDC5_EXTIN_H_POS_CRYCBY when YCbCr422 is selected as the external input format (inp_format).
2.3.4 R_VDC5_SyncControl

Synopsis  Synchronization control setup

Header     r_vdc5.h

Declaration vdc5_error_t R_VDC5_SyncControl(
              const vdc5_channel_t  ch,
              const vdc5_sync_ctrl_t * const param);

Arguments
   • vdc5_channel_t ch: Channel
     — VDC5_CHANNEL_0: Channel 0
     — VDC5_CHANNEL_1: Channel 1
   • vdc5_sync_ctrl_t * param: Synchronization control parameter
     Do not specify NULL.

Return value
   • vdc5_error_t: Error code
     — VDC5_OK: Normal termination
     — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
     — VDC5_ERR_PARAM_NULL: NULL specification error
     — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
     — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
     — VDC5_ERR_RESOURCE_CLK: Clock resource error
     — VDC5_ERR_RESOURCE_INPUT: Input signal resource error

Details
(1) Function
   This function performs the following synchronization control processing:
   • Selects the vertical sync signal.
   • Sets up the period of the sync signal.
   • Sets up the delay of the vertical sync signal.
   • Sets up the full-screen enable signal.
   • Sets up the compensation for the vertical sync signal.

   The settings established by this function remain valid until a hardware reset occurs or they are overwritten by this
function with other settings.

(2) Use conditions
   Before this function is used, the panel clock for the channel designated by ch needs to have been set up. The function
returns a clock resource error (VDC5_ERR_Resource_CLK) if the panel clock is not set up.

   When selecting the external input Vsync signal as the Vsync signal output select to be specified in this function, it is
necessary to enable the video input by calling the function R_VDC5_VideoInput before using this function. The
function returns an input signal resource error (VDC5_ERRRESOURCE_INPUT) if the video input is disabled. For
details, see 2.3.4(3) in a later section.

(3) Parameter details
   The members of the vdc5_sync_ctrl_t structure are described below.

   typedef struct
```c
{
    vdc5_onoff_t                res_vs_sel;
    vdc5_res_vs_in_sel_t        res_vs_in_sel;
    uint16_t                    res_fv;
    uint16_t                    res_fh;
    uint16_t                    res_vsdly;
    vdc5_period_rect_t          res_f;
    const vdc5_vsync_cpmpe_t  * vsync_cpmpe;
} vdc5_sync_ctrl_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_onoff_t</td>
<td>res_vs_sel</td>
<td>Vsync signal output select (free-running Vsync signal)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_OFF: External input Vsync signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_ON: Internally generated free-running Vsync signal</td>
</tr>
<tr>
<td>vdc5_res_vs_in_sel_t</td>
<td>res_vs_in_sel</td>
<td>Horizontal/vertical sync signal output and full-screen enable signal select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_RES_VS_IN_SEL_SC0 (0): Scaler 0 outputs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_RES_VS_IN_SEL_SC1 (1): Scaler 1 outputs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is referenced when cascaded connection is disabled. When cascaded connection is enabled, it is automatically set to VDC5_RES_VS_IN_SEL_SC0 by the VDC5 driver.</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_fv</td>
<td>Free-running Vsync period setting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Free-running Vsync period = (res_fv + 1) x horizontal period [usec]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_fh</td>
<td>Hsync period setting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hsync period [usec] = (res_fh + 1) / pixel clock frequency [MHz]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_vsdly</td>
<td>Vsync signal delay control</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Adjusts the Vsync signal delay in the output Hsync period units. 0 to 255</td>
</tr>
<tr>
<td>vdc5_period_rect_t</td>
<td>res_f</td>
<td>Full-screen enable signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.2.3(1) for the structure. res_f_vs should be 4 lines or more and res_f_vs + res_f_vw should be equal to or less than 2039 lines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>res_f_hs should be 16 clock cycles or more and res_f_hs + res_f_hw should be equal to or less than 2015 clock cycles.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See also Figure 2-6 and its explanation for these settings.</td>
</tr>
<tr>
<td>const vdc5_vsync_cpmpe_t *</td>
<td>vsync_cpmpe</td>
<td>Vsync signal compensation parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specifying NULL turns off the repeated Vsync signal masking control and the compensation of missing Vsync signals.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the compensation of missing Vsync signals is set to OFF, the missing-sync compensating pulse output wait time is set to its maximum value (0xFFFF) by the driver.</td>
</tr>
</tbody>
</table>

For the external input Vsync signal to be selected as the Vsync signal output select (res_vs_sel), the corresponding video input signal needs to have already been set up. The corresponding video input signal is determined as follows:

- When the specified channel (ch) is channel 0 (VDC5_CHANNEL_0):
  - Channel 0 video input if the horizontal/vertical sync signal output and full-screen enable signal select
(res_vs_in_sel) is set to scaler 0 output (VDC5_RES_VS_IN_SEL_SC0)
— Channel 1 video input if the horizontal/vertical sync signal output and full-screen enable signal select
(res_vs_in_sel) is set to scaler 1 output (VDC5_RES_VS_IN_SEL_SC1)
- When the specified channel (ch) is channel 1 (VDC5_CHANNEL_1):
  — Channel 1 video input if the horizontal/vertical sync signal output and full-screen enable signal select
    (res_vs_in_sel) is set to scaler 0 output (VDC5_RES_VS_IN_SEL_SC0)
  — Channel 0 video input if the horizontal/vertical sync signal output and full-screen enable signal select
    (res_vs_in_sel) is set to scaler 1 output (VDC5_RES_VS_IN_SEL_SC1)

Figure 2-6 shows the valid period of the full-screen enable and the output image. The 16 clock cycles before and after
the Hsync signal and the 4 lines before and after the Vsync signal are not included in the valid period of the image.

```
typedef struct
{
    uint16_t res_vmask;
    uint16_t res_vlack;
} vdc5_vsync_cpmpe_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_vsync_cpmpe_t structure are described below.</td>
<td></td>
</tr>
<tr>
<td>Member Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_vmask</td>
</tr>
<tr>
<td></td>
<td>Repeated Vsync signal masking period</td>
</tr>
<tr>
<td></td>
<td>Masking period [usec] = res_vmask x 128 / pixel clock frequency [MHz]</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_vlack</td>
</tr>
<tr>
<td></td>
<td>Missing-sync compensating pulse output wait time</td>
</tr>
<tr>
<td></td>
<td>Wait time [usec] = res_vlack x 128 / pixel clock frequency [MHz]</td>
</tr>
</tbody>
</table>
2.3.5 R_VDC5_DisplayOutput

Synopsis
Display output setup

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_DisplayOutput(
    const vdc5_channel_t  ch,
    const vdc5_output_t  * const param);

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_output_t * param: Display output configuration parameter
  Do not specify NULL.

Return value
• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_NULL: NULL specification error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  — VDC5_ERR_PARAM_CONDITION: Unauthorized condition error
  — VDC5_ERR_RESOURCE_CLK: Clock resource error
  — VDC5_ERR_RESOURCE_VSYNC: Vertical sync signal resource error

Details
(1) Function
This function performs the following processing on the display output:
• Sets up the timing signals for driving the LCD panel.
• Sets up the phase, data sequence, and format of the LCD panel output data.
• Sets up the background color.

The settings established by this function remain valid until a hardware reset occurs or they are overwritten by this function with other settings.

(2) Use conditions
Before this function is used, the panel clock and sync signals for the channel designated by ch need to have been set up. The function returns a clock resource error (VDC5_ERRRESOURCE_CLK) if the panel clock is not set up and a vertical sync signal resource error (VDC5_ERRRESOURCE_VSYNC) if no sync signal is set up.

(3) Parameter details
The members of the vdc5_output_t structure are described below.

typedef struct
{
    uint16_t tcon_half;
    uint16_t tcon_offset;
    const vdc5_lcd_tcon_timing_t * outctrl[VDC5_LCD_TCONSIGNUM];
    vdc5_edge_t outcnt_lcd_edge;
} vdc5_output_t;
```c
vdc5_onoff_t out_endian_on;
vdc5_onoff_t out_swap_on;
vdc5_lcd_outformat_t out_format;
vdc5_lcd_clkfreqsel_t out_frq_sel;
vdc5_lcd_scan_t out_dir_sel;
vdc5_lcd_clkphase_t out_phase;
uint32_t bg_color;
} vdc5_output_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>tcon_half</td>
<td>1/2fH timing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specifies the clock count from the rising edge of the Hsync</td>
</tr>
<tr>
<td></td>
<td></td>
<td>signal as the counting timing of horizontal counter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td>uint16_t</td>
<td>tcon_offset</td>
<td>Offset Hsync signal timing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sets the clock cycle count from the rising edge of the Hsync signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td>const vdc5_lcd_tcon_timing_t*</td>
<td>outctrl[VDC5_LCD_TCONSIG_NUM]</td>
<td>LCD TCON timing signal parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL for any signals that are not to be used.</td>
</tr>
<tr>
<td>vdc5_edge_t</td>
<td>outcnt_lcd_edge</td>
<td>Output phase control of LCD_DATA23 to LCD_DATA0 pin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_EDGE_RISING: Output at the rising edge of LCD_CLK pin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_EDGE_FALLING: Output at the falling edge of LCD_CLK pin.</td>
</tr>
<tr>
<td>vdc5_onoff_t</td>
<td>out_endian_on</td>
<td>Bit endian change on/off control</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_ON</td>
</tr>
<tr>
<td>vdc5_onoff_t</td>
<td>out_swap_on</td>
<td>B/R signal swap on/off control</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_ON</td>
</tr>
<tr>
<td>vdc5_lcd_outformat_t</td>
<td>out_format</td>
<td>Output format select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_OUTFORMAT_RGB888 (0): RGB888</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_OUTFORMAT_RGB666 (1): RGB666</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_OUTFORMAT_RGB565 (2): RGB565</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_OUTFORMAT_SERIAL_RGB (3): Serial RGB</td>
</tr>
<tr>
<td>vdc5_lcd_clkfreqsel_t</td>
<td>out_frq_sel</td>
<td>Clock frequency control</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_PARALLEL_CLKFRQ_1 (0): 100% speed (parallel RGB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_SERIAL_CLKFRQ_3 (1): Triple speed (serial RGB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_SERIAL_CLKFRQ_4 (2): Quadruple speed (serial RGB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is referenced only when out_format is set to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VDC5_LCD_OUTFORMAT_SERIAL_RGB. In this case, setting this parameter to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100% speed is inhibited.</td>
</tr>
<tr>
<td>vdc5_lcd_scan_t</td>
<td>out_dir_sel</td>
<td>Scan direction select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_SERIAL_SCAN_FORWARD (0): Forward scan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_SERIAL_SCAN_REVERSE (1): Reverse scan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is referenced only when out_format is set</td>
</tr>
</tbody>
</table>
### vdc5_lcd_clkphase_t out_phase

Clock phase adjustment during serial RGB output
- VDC5_LCD_SERIAL_CLKPHASE_0 (0): 0 [clocks]
- VDC5_LCD_SERIAL_CLKPHASE_1 (1): 1 [clocks]
- VDC5_LCD_SERIAL_CLKPHASE_2 (2): 2 [clocks]
- VDC5_LCD_SERIAL_CLKPHASE_3 (3): 3 [clocks]

This parameter is referenced only when out_format is set to VDC5_LCD_OUTFORMAT_SERIAL_RGB. It is inhibited to set this parameter to VDC5_LCD_SERIAL_CLKPHASE_3 when out_frq_sel is set to VDC5_LCD_SERIAL_CLKFRQ_3.

### uint32_t bg_color

Background Color
Specify in the RGB888 format (LSB justified).

### vdc5_lcd_tcon_sigsel_t

is an enumeration type for representing the timing signals (LCD TCON) for driving the LCD panel.

```c
typedef enum {
    VDC5_LCD_TCONSIG_STVA_VS = 0,
    VDC5_LCD_TCONSIG_STVB_VE,
    VDC5_LCD_TCONSIG_STH_SP_HS,
    VDC5_LCD_TCONSIG_STB_LP_HE,
    VDC5_LCD_TCONSIG_CPV_GCK,
    VDC5_LCD_TCONSIG_POLA,
    VDC5_LCD_TCONSIG_POLB,
    VDC5_LCD_TCONSIG_DE,
    VDC5_LCD_TCONSIG_NUM
} vdc5_lcd_tcon_sigsel_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_LCD_TCONSIG_STVA_VS</td>
<td>0</td>
<td>Gate start signal, Vsync signal (STVA/VS)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_STVB_VE</td>
<td>1</td>
<td>Gate start signal, vertical enable signal (STVB/VE)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_STH_SP_HS</td>
<td>2</td>
<td>Source start signal, Hsync signal (STH/SP/HS)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_STB_LP_HE</td>
<td>3</td>
<td>Source strobe signal, horizontal enable signal (STB/LP/HE)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_CPV_GCK</td>
<td>4</td>
<td>Gate clock signal (CPV/GCK)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_POLA</td>
<td>5</td>
<td>VCOM voltage polarity control signal (POLA)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_POLB</td>
<td>6</td>
<td>VCOM voltage polarity control signal (POLB)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_DE</td>
<td>7</td>
<td>Data enable signal (DE)</td>
</tr>
<tr>
<td>VDC5_LCD_TCONSIG_NUM</td>
<td>8</td>
<td>Number of LCD panel drive signal types</td>
</tr>
</tbody>
</table>

The members of the `vdc5_lcd_tcon_timing_t` structure are described below.

```c
typedef struct {
    uint16_t                tcon_hsvs;
    uint16_t                tcon_hwvw;
    vdc5_lcd_tcon_polmode_t tcon_md;
    vdc5_lcd_tcon_refsel_t  tcon_hs_sel;
    vdc5_sig_pol_t          tcon_inv;
    vdc5_lcd_tcon_pin_t     tcon_pin;
    vdc5_edge_t             outcnt_edge;
} vdc5_lcd_tcon_timing_t;
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>tcon_hsvs</td>
<td>Signal pulse start position (first changing timing)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starts pulse output after the time specified by the value of tcon_hsvs from the rising edge of the reference signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF [clock cycles, 1/2fH cycles]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set a value of 1 or greater if tcon_md is set to a value other than VDC5_LCD_TCON_POLMD_NORMAL when using the POLA/POLB signal.</td>
</tr>
<tr>
<td>uint16_t</td>
<td>tcon_hwvw</td>
<td>Pulse width (second changing timing)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outputs a pulse of the duration of the value of tcon_hwvw.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF [clock cycles, 1/2fH cycles]</td>
</tr>
<tr>
<td>vdc5_lcd_tcon_polmode_t</td>
<td>tcon_md</td>
<td>POLA/POLB signal generation mode select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_POLMD_NORMAL (0): Normal mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Generates the signal whose polarity is inverted every horizontal period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_POLMD_1X1REV (1): 1x1 reverse mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Generates the signal whose polarity is inverted every horizontal period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_POLMD_1X2REV (2): 1x2 reverse mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Generates the signal whose polarity is inverted in the first horizontal period and is subsequently inverted every two horizontal periods.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_POLMD_2X2REV (3): 2x2 reverse mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Generates the signal whose polarity is inverted every two horizontal periods.</td>
</tr>
<tr>
<td>vdc5_lcd_tcon_refsel_t</td>
<td>tcon_hs_sel</td>
<td>Operating reference select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_REFSEL_HSYNC (0): Hsync signal reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_REFSEL_OFFSET_H (1): Offset Hsync signal reference</td>
</tr>
<tr>
<td>vdc5_sig_pol_t</td>
<td>tcon_inv</td>
<td>Polarity inversion control of signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_SIG_POL_NOT_INVERTED: Not inverted (positive polarity)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_SIG_POL_INVERTED: Inverted (negative polarity)</td>
</tr>
<tr>
<td>vdc5_lcd_tcon_pin_t</td>
<td>tcon_pin</td>
<td>LCD TCON output pin select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_NON (-1): Nothing output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_0 (0): LCD_TCON0 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_1 (1): LCD_TCON1 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_2 (2): LCD_TCON2 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_3 (3): LCD_TCON3 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_4 (4): LCD_TCON4 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_5 (5): LCD_TCON5 output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_LCD_TCON_PIN_6 (6): LCD_TCON6 output.</td>
</tr>
<tr>
<td>vdc5_edge_t</td>
<td>outcnt_edge</td>
<td>Output phase control of the signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_EDGE_RISING: Output at the rising edge of LCD_CLK pin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_EDGE_FALLING: Output at the falling edge of LCD_CLK pin.</td>
</tr>
</tbody>
</table>
Different members of the `vdc5_lcd_tcon_timing_t` structure are referenced depending on the type of the LCD panel drive signals that are to be set up. The table below summarizes the members that are valid or invalid when the respective signals are set up. Invalid members are not referenced.

**Table 2-9  Valid Parameters for the LCD Panel Drive Signals**

<table>
<thead>
<tr>
<th>LCD Panel Drive Signal</th>
<th>vdc5_lcd_tcon_timing_t Structure Member</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>tcon_hsvs</td>
</tr>
<tr>
<td>STVA/VS</td>
<td>valid</td>
</tr>
<tr>
<td>STVB/VE</td>
<td>valid</td>
</tr>
<tr>
<td>STH/SP/HS</td>
<td>valid</td>
</tr>
<tr>
<td>STB/LP/HE</td>
<td>valid</td>
</tr>
<tr>
<td>CPV/GCK</td>
<td>valid</td>
</tr>
<tr>
<td>POLA</td>
<td>valid</td>
</tr>
<tr>
<td>POLB</td>
<td>valid</td>
</tr>
<tr>
<td>DE</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: 'valid' denotes a valid member whose value is referenced. ' - ' denotes an invalid member whose value is not referenced.
2.3.6 R_VDC5_CallbackISR

Synopsis
Interrupt callback setup

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_CallbackISR(
    const vdc5_channel_t ch,
    const vdc5_int_t * const param);

Arguments
- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_int_t * param: Interrupt callback setup parameter
  Do not specify NULL.

Return value
- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  - VDC5_ERR_RESOURCE_CLK: Clock resource error
  - VDC5_ERR_RESOURCE_VSYNC: Vertical sync signal resource error

Details

(1) Function
This function performs the following VDC5 interrupt processing:
- Enables the interrupt when the pointer to the corresponding interrupt callback function is specified.
- Registers the specified interrupt callback function.
- Disables the interrupt when the pointer to the corresponding interrupt callback function is not specified.

With this driver, all of the VDC5 interrupts are disabled and the registered entries of the callback functions are removed during R_VDC5_Initialize and R_VDC5_Terminate. The setting of an existing interrupt can also be overwritten by applying this function R_VDC5_CallbackISR to that interrupt.

(2) Use conditions
Before this function is used, the panel clock and sync signals for the channel designated by ch need to have been set up. The function returns a clock resource error (VDC5_ERR_RESOURCE_CLK) if the panel clock is not set up and a vertical sync signal resource error (VDC5_ERR_RESOURCE_VSYNC) if no sync signal is set up.

(3) Parameter details
The members of the vdc5_int_t structure are described below.

```c
typedef struct
{
    vdc5_int_type_t type;
    void (* callback)(vdc5_int_type_t);
    uint16_t line_num;
} vdc5_int_t;
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| vdc5_int_type_t | type | VDC5 interrupt type  
See 2.2.2(8) for details. |
| void (* callback)(vdc5_int_type_t) | | Interrupt callback function pointer  
Specify the pointer to the interrupt callback function  
associated with the interrupt that is specified in type. The  
callback function needs to be implemented by the user.  
The interrupt processing of the interrupt for which a  
callback function is specified becomes enabled. If a '0' is  
specified in callback, the interrupt specified in type  
becomes disabled. |
| uint16_t | line_num | Line interrupt set  
An interrupt signal is output when the line position of the  
image matches the value of line_num.  
This parameter is valid only when one of the following line  
interrupts is specified in type:  
- VDC5_INT_TYPE_VLINE  
- VDC5_INT_TYPE_S0_WLINE  
- VDC5_INT_TYPE_S1_WLINE  
- VDC5_INT_TYPE_OIR_VLINE |
2.3.7 R_VDC5_WriteDataControl

Synopsis
Data write control processing

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_WriteDataControl(
    const vdc5_channel_t ch,
    const vdc5_layer_id_t layer_id,
    const vdc5_write_t * const param);

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_layer_id_t layer_id: Layer ID
  — VDC5_LAYER_ID_0_WR: Layer 0 write processing
  — VDC5_LAYER_ID_1_WR: Layer 1 write processing
  — VDC5_LAYER_ID_OIR_WR: OIR layer write processing
• vdc5_write_t * param: Data write control parameter
  Do not specify NULL.

Return value
• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  — VDC5_ERR_PARAM_NULL: NULL specification error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  — VDC5_ERRRESOURCE_INPUT: Input signal resource error
  — VDC5_ERRRESOURCE_LAYER: Layer resource error

Details

(1) Function
This function performs the following data write control processing:

• Sets up the input image area to be captured.
• Makes input image scaling-down and rotation control settings (layers 0 and 1 only).
• Makes frame buffer write control settings.

(2) Use conditions
Before using this function for a layer other than the OIR layer, it is necessary to enable a video input by calling the function R_VDC5_VideoInput. For layer 0, the video input for the same channel is required. For layer 1, the video input for another channel is required. If no video input is enabled, the function returns an input signal resource error (VDC5_ERRRESOURCE_INPUT).

This function returns a layer resource error (VDC5_ERRRESOURCE_LAYER) if the layer that is specified in layer_id is found enabled already when this function is used. The enabled layer can be disabled by calling the function R_VDC5_ReleaseDataControl.

The write processing for layer 0 and the write processing for layer 1 of another channel can reference the same video input. The video input for channel 0, for example, is available for the channel 0 layer 0 write processing and the channel...
1 layer 1 write processing. In this case, it is necessary to specify the same frame buffer video-signal writing format for the two processes of layer write processing. This is a restriction imposed by the fact that only one color matrix that can be used for color conversion when the video input data is written into the frame buffer.

Control of the vertical scale-down processing which is available for layers 0 and 1 is mutually exclusive with the control of the vertical scale-up processing. Normal operation of the driver operation is not guaranteed if vertical scale-down and vertical scale-up are specified at the same time. The setup of vertical scale-up processing is accomplished by the functions R_VDC5_ReadDataControl and R_VDC5_ChangeReadProcess.

(3) Parameter details

The members of the `vdc5_write_t` structure are described below:

```c
typedef struct {
    vdc5_scalingdown_rot_t      scalingdown_rot;
    vdc5_wr_rd_swa_t            res_wrswa;
    vdc5_res_md_t               res_md;
    vdc5_bst_md_t               res_bst_md;
    vdc5_res_inter_t            res_inter;
    vdc5_res_fs_rate_t          res_fs_rate;
    vdc5_res_fld_sel_t          res_fld_sel;
    vdc5_onoff_t                res_dth_on;
    void                        * base;
    uint32_t                    ln_off;
    uint32_t                    fml_num;
    uint32_t                    fml_off;
    void                        * btm_base;
} vdc5_write_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_scalingdown_rot_t</td>
<td>scalingdown_rot</td>
<td>Scaling-down and rotation parameter</td>
</tr>
<tr>
<td>vdc5_wr_rd_swa_t</td>
<td>res_wrswa</td>
<td>8-bit, 16-bit, or 32-bit swap setting</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_NON (0): 1-2-3-4-5-6-7-8</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Not swapped</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_8BIT (1): 2-1-4-3-6-5-8-7</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 8-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_16BIT (2): 3-4-1-2-7-8-5-6</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 16-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_16_BIT (3): 4-3-2-1-8-7-6-5</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 16-bit units + 8-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_32BIT (4): 5-6-7-8-1-2-3-4</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 32-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_32_8BIT (5): 6-5-8-7-2-1-4-3</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 32-bit units + 8-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_32_16BIT (6): 7-8-5-6-3-4-1-2</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 32-bit units + 16-bit units</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• VDC5_WR_RD_WRSWA_32_16_8BIT (7): 8-7-6-5-4-3-2-1</td>
</tr>
<tr>
<td></td>
<td>res_wrswa</td>
<td>• Swapped in 32-bit units + 16-bit units + 8-bit units</td>
</tr>
<tr>
<td>vdc5_res_md_t</td>
<td>res_md</td>
<td>Frame buffer video-signal writing format</td>
</tr>
<tr>
<td></td>
<td>res_md</td>
<td>• VDC5_RES_MD_YCBCR422 (0): YCbCr422</td>
</tr>
<tr>
<td></td>
<td>res_md</td>
<td>• VDC5_RES_MD_RGB565 (1): RGB565</td>
</tr>
<tr>
<td></td>
<td>res_md</td>
<td>• VDC5_RES_MD_RGB888 (2): RGB888</td>
</tr>
</tbody>
</table>
• **VDC5_RES_MD_YCBCR444** (3): YCbCr444

**vdc5_bst_md_t**  
res_bst_md  
Transfer burst length for frame buffer writing  
• **VDC5_BST_MD_32BYTE** (0): 32-byte transfer (4 bursts)  
• **VDC5_BST_MD_128BYTE** (1): 128-byte transfer (16 bursts)

**vdc5_res_inter_t**  
res_inter  
Field operating mode select  
• **VDC5_RES_INTER_PROGRESSIVE** (0): Progressive  
• **VDC5_RES_INTER_INTERLACE** (1): Interlace

**vdc5_res_fs_rate_t**  
res_fs_rate  
Writing rate  
• **VDC5_RES_FS_RATE_PER1** (0): 1/1 an input signal  
• **VDC5_RES_FS_RATE_PER2** (1): 1/2 an input signal  
• **VDC5_RES_FS_RATE_PER4** (2): 1/4 an input signal  
• **VDC5_RES_FS_RATE_PER8** (3): 1/8 an input signal

**vdc5_res_fld_sel_t**  
res_fld_sel  
Write field select  
This parameter is valid only when res_fs_rate is set to a value other than VDC5_RES_FS_RATE_PER1.  
• **VDC5_RES_FLD_SEL_TOP** (0): Top field  
• **VDC5_RES_FLD_SEL_BOTTOM** (1): Bottom field

**vdc5_onoff_t**  
res_dth_on  
Dither correction on/off  
• **VDC5_OFF**: Rounded off  
• **VDC5_ON**: 2x2 dither pattern

**void **  
base  
Frame buffer base address  
Do not specify NULL.  
When the value specified in res_bst_md is:  
• **VDC5_BST_MD_32BYTE**  
  Specify an address that is aligned to 32 bytes.  
• **VDC5_BST_MD_128BYTE**  
  Specify an address that is aligned to 128 byte.

**uint32_t**  
ln_off  
Frame buffer line offset address  
0x0000 to 0x7FFF  
When the value specified in res_bst_md is:  
• **VDC5_BST_MD_32BYTE**  
  Specify a multiple of 32.  
• **VDC5_BST_MD_128BYTE**  
  Specify a multiple of 128.

**uint32_t**  
flm_num  
Number of frames of buffer to be written to  
0x0000 to 0x03FF  
Number of frames defined by flm_num + 1 is used.  
Specify 2 frames ("1") or more for rotation processing.

**uint32_t**  
flm_off  
Frame buffer frame offset address  
0x00000000 to 0x00FFFF  
This parameter is invalid when the number of frames is 1 (flm_num is set to '0').  
When the value specified in res_bst_md is:  
• **VDC5_BST_MD_32BYTE**  
  Specify a multiple of 32.  
• **VDC5_BST_MD_128BYTE**  
  Specify a multiple of 128.

**void **  
btm_base  
Frame buffer base address for bottom  
Specify NULL if not required.  
When the value specified in res_bst_md is:  
• **VDC5_BST_MD_32BYTE**
Specify an address that is aligned to 32 bytes.
- VDC5_BST_MD_128BYTE
  Specify an address that is aligned to 128 bytes.

The members of the vdc5_scalingdown_rot_t structure are described below.

typedef struct
{
  vdc5_period_rect_t  res;
  vdc5_onoff_t       res_pfil_sel;
  uint16_t            res_out_vw;
  uint16_t            res_out_hw;
  vdc5_onoff_t        adj_sel;
  vdc5_wr_md_t        res_ds_wr_md;
} vdc5_scalingdown_rot_t;

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_period_rect_t</td>
<td>res</td>
<td>Image area to be captured</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.2.3(1) for the vdc5_period_rect_t structure.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>res.vs should be 4 lines or more and res.vs + res.vw should be equal to or less than 2039 lines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>res.hs should be 16 clock cycles or more and res.hs + res.hw should be equal to or less than 2015 clock cycles.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The actual vertical position setting for video signal capturing is res.vs + 1.</td>
</tr>
<tr>
<td>vdc5_onoff_t</td>
<td>res_pfil_sel</td>
<td>Prefilter mode select for brightness signals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_ON</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_out_vw</td>
<td>Number of valid lines in vertical direction output by scaling-down control block (lines)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify a value that is aligned in 4-line units and equal to or smaller than the res.vw.value.</td>
</tr>
<tr>
<td>uint16_t</td>
<td>res_out_hw</td>
<td>Number of valid horizontal pixels output by scaling-down control block (video-image clock cycles)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify a value that is aligned in 4-pixel units and equal to or smaller than the res.hw.value.</td>
</tr>
<tr>
<td>vdc5_onoff_t</td>
<td>adj_sel</td>
<td>Handling for lack of last-input line</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specifies whether to take countermeasures for decreasing the influence by the lack of last-input line in scale-down processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_ON</td>
</tr>
<tr>
<td>vdc5_wr_md_t</td>
<td>res_ds_wr_md</td>
<td>Frame buffer writing mode for image processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_WR_MD_NORMAL (0): Normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_WR_MD_MIRROR (1): Horizontal mirroring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_WR_MD_ROT_90DEG (2): 90-degree rotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_WR_MD_ROT_180DEG (3): 180-degree rotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- VDC5_WR_MD_ROT_270DEG (4): 270-degree rotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Setting this parameter to 90-degree, 180-degree, or 270-degree rotation is valid only when frame buffer video-signal writing format (res_md) is set to YCbCr422 or RGB565.</td>
</tr>
</tbody>
</table>
Figure 2-7 shows the relationship between the settings of the frame buffer related parameters and the memory allocation.

**A. Memory Map**

- **base**: Frame buffer base address
- **flm_off**: Frame buffer frame offset address [bytes]
- **n**: Number of frame buffers (= flm_num + 1)
- **height**: Number of valid lines in vertical direction output by scaling-down control block [lines]
- **width**: Number of valid horizontal pixels output by scaling-down control block [video-image clock cycles]
- **ln_off**: Frame buffer line offset address [bytes]

**B. Data Arrangement**

The width and height of the image in the frame buffer (width and height in Figure 2-7) are the number of valid horizontal pixels (res_out_hw) and the number of valid lines in vertical direction (res_out_vw) output by scaling-down control block, respectively. But when the frame buffer writing mode for image processing (res_ds_wr_md) is set to 90-degree rotation or 270-degree rotation, the width and height of the image are res_out_vw and res_out_hw, respectively.
2.3.8 R_VDC5_ChangeWriteProcess

Synopsis Data write change processing

Header r_vdc5.h

Declaration
vdc5_error_t R_VDC5_ChangeWriteProcess(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id,
    const vdc5_write_chg_t * const param);

Arguments
- vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
- vdc5_layer_id_t layer_id: Layer ID
  — VDC5_LAYER_ID_0_WR: Layer 0 write processing
  — VDC5_LAYER_ID_1_WR: Layer 1 write processing
  — VDC5_LAYER_ID_OIR_WR: OIR layer write processing
- vdc5_write_chg_t * param: Data write change parameter
  Do not specify NULL.

Return value
- vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  — VDC5_ERR_PARAM_NULL: NULL specification error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  — VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

(1) Function

This function takes the following data write control actions during data write processing:
- Changes the input image area to be captured.
- Makes changes with respect to scaling-down and rotation control of the input image (layers 0 and 1 only).

(2) Use conditions

This function returns a layer resource error (VDC5_ERRRESOURCE_LAYER) unless the layer specified in layer_id of this function meets the following conditions:
- The specified layer is enabled.
- The specified layer is running.

A layer is enabled by calling the function R_VDC5_WriteDataControl. A layer that is in the stopped state can be started by calling the function R_VDC5_StartProcess.

Control of the vertical scale-down processing which is available for layers 0 and 1 is mutually exclusive with the control of the vertical scale-up processing. Normal operation of the driver operation is not guaranteed if vertical scale-down and vertical scale-up are specified at the same time. The setup of vertical scale-up processing is accomplished by the functions R_VDC5_ReadDataControl and R_VDC5_ChangeReadProcess.
(3) Parameter details

The members of the `vdc5_write_chg_t` structure are described below.

```c
typedef struct {
    vdc5_scalingdown_rot_t scalingdown_rot;
} vdc5_write_chg_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_scalingdown_rot_t</td>
<td>scalingdown_rot</td>
<td>Scaling-down and rotation parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.3.7(3) for details.</td>
</tr>
</tbody>
</table>

This function can be used to change the rotation control mode during data write processing. The size of the frame buffer that is required for the data write may be altered if the rotation angle is changed by 90 or 270 degrees from the current angle.

Figure 2-8 shows two cases of processing in which the frame buffer size is altered as the result of rotating the input image by 90 degrees. Before the rotation, the width (hw) and height (vw) of the frame buffer that is necessary for data write processing are set to (res_out_hw) and (res_out_vw), respectively. Since these size values change as the result of the 90-degree rotation, data is likely to be written into an unexpected area. To avoid this situation, it is necessary to reserve a larger frame buffer in advance ((a) and (b) in the figure).
### 2.3.9 R_VDC5_ReadDataControl

**Synopsis**  
Data read control processing

**Header**  
r_vdc5.h

**Declaration**  
```c
vdc5_error_t R_VDC5_ReadDataControl(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id,
    const vdc5_read_t  * const param);
```

**Arguments**  
- `vdc5_channel_t ch`: Channel  
  - VDC5_CHANNEL_0: Channel 0  
  - VDC5_CHANNEL_1: Channel 1  
- `vdc5_layer_id_t layer_id`: Layer ID  
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing  
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing  
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing  
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing  
  - VDC5_LAYER_ID_OIR_RD: OIR layer read processing  
- `vdc5_read_t * param`: Data read control parameter  
  Do not specify NULL.

**Return value**  
- `vdc5_error_t`: Error code  
  - VDC5_OK: Normal termination  
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error  
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error  
  - VDC5_ERR_PARAM_NULL: NULL specification error  
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error  
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error  
  - VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error  
  - VDC5_ERR_PARAM_CONDITION: Unauthorized condition error  
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

**Details**

(1) **Function**

This function performs the following data read control processing:

- Sets up the display area for graphics images.
- Makes image scale-up control settings (layers 0 and 1 only).
- Makes frame buffer read control settings.

(2) **Use conditions**

This function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the layer specified in `layer_id` is found enabled already when this function is used. The enabled layer can be disabled by calling the function R_VDC5_ReleaseDataControl.

Control of the vertical scale-up processing which is available for layers 0 and 1 is mutually exclusive with the control of the vertical scale-down processing. Normal operation of the driver operation is not guaranteed if vertical scale-down and vertical scale-up are specified at the same time. The setup of vertical scale-down processing is accomplished by the functions R_VDC5_WriteDataControl and R_VDC5_ChangeWriteProcess.
(3) Parameter details

The `vdc5_read_t` structure is described below.

```c
typedef struct {
vdc5_gr_ln_off_dir_t            gr_ln_off_dir;
vdc5_gr_flm_sel_t               gr_flm_sel;
vdc5_onoff_t                    gr_imr_flm_inv;
vdc5_bst_md_t                   gr_bst_md;
void                          * gr_base;
uint32_t                        gr_ln_off;
const vdc5_width_read_fb_t    * width_read_fb;
vdc5_onoff_t                    adj_sel;
vdc5_gr_format_t                gr_format;
vdc5_gr_ycc_swap_t              gr_ycc_swap;
vdc5_wr_rd_swa_t                gr_rdswa;
  vdc5_period_rect_t            gr_grc;
} vdc5_read_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_gr_ln_off_dir_t</td>
<td>gr_ln_off_dir</td>
<td>Line offset address direction of the frame buffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_LN_OFF_DIR_INC (0): Increments the address by the line offset address.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_LN_OFF_DIR_DEC (1): Decrements the address by the line offset address.</td>
</tr>
<tr>
<td>vdc5_gr_flm_sel_t</td>
<td>gr_flm_sel</td>
<td>Frame buffer address setting signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_FLM_SEL_SCALE_DOWN (0): Links to scaling-down process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_FLM_SEL_FLM_NUM (1): Selects frame 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_FLM_SEL_DISTORTION (2): Links to distortion correction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_GR_FLM_SEL_POINTER_BUFF (3): Links to pointer buffer.</td>
</tr>
<tr>
<td>vdc5_onoff_t</td>
<td>gr_imr_flm_inv</td>
<td>Frame buffer number for distortion correction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_OFF: Does not replace the numbers of the frames to be read.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_ON: Replaces the numbers of the frames to be read.</td>
</tr>
<tr>
<td>vdc5_bst_md_t</td>
<td>gr_bst_md</td>
<td>Frame buffer burst transfer mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_BST_MD_32BYTE (0): 32-byte transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_BST_MD_128BYTE (1): 128-byte transfer</td>
</tr>
<tr>
<td>void *</td>
<td>gr_base</td>
<td>Frame buffer base address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Do not specify NULL if <code>gr_flm_sel</code> is set to a value other than</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VDC5_GRL_FLM_SEL_POINTER_BUFF.</td>
</tr>
<tr>
<td>uint32_t</td>
<td>gr_ln_off</td>
<td>Frame buffer line offset address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x7FFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When the value specified in <code>gr_bst_md</code> is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_BST_MD_32BYTE Specify a multiple of 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VDC5_BST_MD_128BYTE Specify a multiple of 128.</td>
</tr>
<tr>
<td>Variable Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>const vdc5_width_read_fb_t *</td>
<td>Size of the frame buffer to be read. If NULL is specified, the size values of the frame buffer are assumed to be equal to the graphics display area width (gr_grc.hw) and height (gr_grc.vw).</td>
<td></td>
</tr>
<tr>
<td>vdc5_onoff_t adj_sel</td>
<td>Folding handling. Specifies whether to take countermeasures for decreasing the influence by folding pixels in scale-up processing.</td>
<td></td>
</tr>
<tr>
<td>vdc5_gr_format_t gr_format</td>
<td>Format of the frame buffer read signal.</td>
<td></td>
</tr>
<tr>
<td>vdc5_gr_ycc_swap_t gr_ycc_swap</td>
<td>Swapping of data read from buffer in the YCbCr422 format. This parameter is valid only when gr_format is set to VDC5_GR_FORMAT_YCBCR422.</td>
<td></td>
</tr>
<tr>
<td>vdc5_wr_rd_swa_t gr_rdswa</td>
<td>8-bit, 16-bit, or 32-bit swap setting.</td>
<td></td>
</tr>
<tr>
<td>vdc5_period_rect_t gr_grc</td>
<td>Graphics display area. See 2.2.3(1) for the vdc5_period_rect_t structure.</td>
<td></td>
</tr>
</tbody>
</table>
gr_grc.vs should be 4 lines or more and gr_grc.vs +
gr_grc.vw should be equal to or less than 2039 lines.
gr_grc.hs should be 16 clock cycles or more and gr_grc.hs +
gr_grc.hw should be equal to or less than 2015 clock
cycles.

Note: YCbCr422 and YCbCr444 can be specified only for graphics 0 and 1.

The legitimate parameter values for the frame buffer address setting signal (gr_flm_sel) differ from layer to layer. See Table 2-10 for legitimate parameter values for the layers.

### Table 2-10  Legitimate Frame Buffer Address Setting Signal Parameter Values

<table>
<thead>
<tr>
<th>Layer ID</th>
<th>Legitimate Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_LAYER_ID_0_RD</td>
<td>VDC5_GR_FLM_SEL_SCALE_DOWN</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_FLM_NUM</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_DISTORTION</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_POINTER_BUFF</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_1_RD</td>
<td>VDC5_GR_FLM_SEL_SCALE_DOWN</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_FLM_NUM</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_POINTER_BUFF</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_2_RD</td>
<td>VDC5_GR_FLM_SEL_FLM_NUM</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_3_RD</td>
<td>VDC5_GR_FLM_SEL_FLM_NUM</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_OIR_RD</td>
<td>VDC5_GR_FLM_SEL_SCALE_DOWN</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_FLM_NUM</td>
</tr>
<tr>
<td></td>
<td>VDC5_GR_FLM_SEL_DISTORTION *</td>
</tr>
</tbody>
</table>

Note: For the OIR layer, IMR-LSD is allowed only for channel 0. Consequently, VDC5_GR_FLM_SEL_DISTORTION can be specified only when channel 0 is to be used.

The members of the vdc5_width_read_fb_t structure are described below.

```c
typedef struct
{
  uint16_t in_vw;
  uint16_t in_hw;
} vdc5_width_read_fb_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>in_vw</td>
<td>Number of lines in a frame (lines)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
<tr>
<td>uint16_t</td>
<td>in_hw</td>
<td>Width of the horizontal valid period (pixels)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0x0000 to 0x07FF</td>
</tr>
</tbody>
</table>

Figure 2-9 shows the relationship between the parameter settings for the frame buffer to be used during data read processing and the memory allocation.
(a), (b): Frame buffer base address (gr_base)
in_vw: Number of lines in a frame [lines]
in.hw: Width of the horizontal valid period [pixels]
gr_ln_off: Frame buffer line offset address [bytes]

Figure 2-9 Read Processing Frame Buffer Related Parameter Settings and Memory Allocation

The value of the frame buffer base address (gr_base) need be changed according to the setting of the line offset address direction of the frame buffer (gr_ln_off_dir).

- When gr_ln_off_dir is set to VDC5_GR_LN_OFF_DIR_INC
  Data is read sequentially starting at the beginning of the frame buffer. Specify in gr_base the start address of the frame buffer (Figure 2-9, (a)).
- When gr_ln_off_dir is set to VDC5_GR_LN_OFF_DIR_DEC
  Data is read from the last line of the frame buffer. Specify in gr_base the address of the last line in the frame buffer (Figure 2-9, (b)).
2.3.10  R_VDC5_ChangeReadProcess

Synopsis  Data read change processing

Header    r_vdc5.h

Declaration  vdc5_error_t R_VDC5_ChangeReadProcess(
                      const vdc5_channel_t       ch,
                      const vdc5_layer_id_t     layer_id,
                      const vdc5_read_chg_t     * param);

Arguments
- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_layer_id_t layer_id: Layer ID
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
  - VDC5_LAYER_ID_OIR_RD: OIR layer read processing
- vdc5_read_chg_t * param: Data read change parameter
  Do not specify NULL.

Return value
- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  - VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details
(1)  Function
This function takes the following data read control actions during data read processing:
- Changes the frame buffer base address.
- Changes the frame buffer read size (image scale-up control) (layers 0 and 1 only).
- Changes the display area for graphics images.
- Changes the graphics display mode.

(2)  Use conditions
This function returns a layer resource error (VDC5_ERRRESOURCE_LAYER) unless the layer specified in layer_id of this function meets the following conditions:
- The specified layer is enabled.
- The specified layer is running.

A layer is enabled by calling the function R_VDC5_ReadDataControl. A layer that is in the stopped state can be started by calling the function R_VDC5_StartProcess.
Control of the vertical scale-up processing which is available for layers 0 and 1 is mutually exclusive with the control of the vertical scale-down processing. Normal operation of the driver operation is not guaranteed if vertical scale-down and vertical scale-up are specified at the same time. The setup of vertical scale-down processing is accomplished by the functions R_VDC5_WriteDataControl and R_VDC5_ChangeWriteProcess.

(3) Parameter details

The members of the **vdc5_read_chg_t** structure are described below.

```c
typedef struct {
    void                          * gr_base;
    const vdc5_width_read_fb_t    * width_read_fb;
    const vdc5_period_rect_t      * gr_grc;
    const vdc5_gr Disp_sel_t      * gr_disp_sel;
} vdc5_read_chg_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>gr_base</td>
<td>Frame buffer base address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value is not to be changed.</td>
</tr>
<tr>
<td>const vdc5_width_read_fb_t *</td>
<td>width_read_fb</td>
<td>Size of the frame buffer to be read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.3.9(3) for the vdc5_width_read_fb_t structure.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value is not to be changed.</td>
</tr>
<tr>
<td>const vdc5_period_rect_t *</td>
<td>gr_grc</td>
<td>Graphics display area</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.2.3(1) for the vdc5_period_rect_t structure.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value is not to be changed.</td>
</tr>
<tr>
<td>const vdc5_gr Disp_sel_t *</td>
<td>gr_disp_sel</td>
<td>Graphics display mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See 2.2.2(9) and 2.3.11(3) for details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value is not to be changed.</td>
</tr>
</tbody>
</table>

The values of the parameters in the **vdc5_read_chg_t** structure are not changed if NULL is specified for the parameters. The parameters in gr_base, width_read_fb and gr_grc retain the values that have been set up by the function R_VDC5_ReadDataControl. The parameter in gr_disp_sel retains the value that has been set up by the function R_VDC5_StartProcess.
2.3.11  R_VDC5_StartProcess

Synopsis  Data write/read start processing

Header  r_vdc5.h

Declaration  

```c
vdc5_error_t R_VDC5_StartProcess(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id,
    const vdc5_start_t  * const param);
```

Arguments  
- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- `vdc5_layer_id_t layer_id`: Layer ID
  - VDC5_LAYER_ID_ALL: All enabled layers
  - VDC5_LAYER_ID_0_WR: Layer 0 write processing
  - VDC5_LAYER_ID_1_WR: Layer 1 write processing
  - VDC5_LAYER_ID_OIR_WR: OIR layer write processing
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
  - VDC5_LAYER_ID_OIR_RD: OIR layer read processing
- `vdc5_start_t * param`: Data write/read start parameter
  Do not specify NULL.

Return value  
- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

1) Function

This function performs layer start processing. If the layer ID specified in `layer_id` is VDC5_LAYER_ID_ALL, the function starts all the layers that are in the stopped state and also enabled. If the layer ID is not VDC5_LAYER_ID_ALL, the function starts only the specified layer.

When performing start processing for write, the function starts a write to the frame buffer. When performing start processing for read, the function starts a read from the frame buffer and sets the graphics display mode to the specified values for each layer.

2) Use conditions

No particular use conditions are imposed if `layer_id` is found to be set to VDC5_LAYER_ID_ALL when the function is used. In the other cases, the conditions listed below apply. If these conditions are not met, the function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER).

- The specified layer is enabled.
- The specified layer is stopped.

Layers involved in write processing are enabled by calling the function R_VDC5_WriteDataControl and layers involved in read processing are enabled by calling the function R_VDC5_ReadDataControl. Layers that are running can be stopped by calling the function R_VDC5_StopProcess.

(3) Parameter details

The members of the vdc5_start_t structure are described below.

```c
typedef struct
{
    const vdc5_gr_disp_sel_t      * gr_disp_sel;
} vdc5_start_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>const vdc5_gr_disp_sel_t*</td>
<td>gr_disp_sel</td>
<td>Graphics display mode See 2.2.2(9) and the following description for details. Specify NULL when this parameter value is not to be changed. This setting pertains to the graphics (read processing). This setting is invalid if a layer for write processing is specified in layer_id.</td>
</tr>
</tbody>
</table>

See the sample settings given below for the setup of gr_disp_sel.

1. When specifying VDC5_LAYER_ID_ALL in layer_id

Specify the graphics display settings for all the graphics (layers for read processing). Specify VDC5_DISPSEL_IGNORED for layers that need no change.

```c
   1  vdc5_error_t    error;
   2  vdc5_gr_disp_sel_t     gr_disp_sel[VDC5_GR_TYPE_NUM];
   3  vdc5_start_t      start;

   4
   5  gr_disp_sel[VDC5_GR_TYPE_GR0] = VDC5_DISPSEL_IGNORED;
   6  gr_disp_sel[VDC5_GR_TYPE_GR1] = VDC5_DISPSEL_IGNORED;
   7  gr_disp_sel[VDC5_GR_TYPE_GR2] = VDC5_DISPSEL_CURRENT;
   8  gr_disp_sel[VDC5_GR_TYPE_GR3] = VDC5_DISPSEL_BLEND;
   9  gr_disp_sel[VDC5_GR_TYPE_VIN] = VDC5_DISPSEL_IGNORED;
  10  gr_disp_sel[VDC5_GR_TYPE_OIR] = VDC5_DISPSEL_IGNORED;
  11
  12  start.gr_disp_sel = gr_disp_sel;
  13
  14  error = R_VDC5_StartProcess(VDC5_CHANNEL_1, VDC5_LAYER_ID_ALL, &start);
```

2. When specifying a single layer in layer_id

Specify the graphics display mode only for the specified graphics (layer for read processing). Given below are example settings for graphics 2 (read process for layer 2).

```c
```
The graphics display mode for the layers are initialized by the driver when calling the function R_VDC5_DisplayOutput, R_VDC5_ReadDataControl, and R_VDC5_StopProcess. If they are not changed through the function R_VDC5_StartProcess, the initial values that are set up by the driver are retained. Table 2-11 shows the initial values of the graphics display mode for the layers.

### Table 2-11 Graphics Display Mode Initial Values

<table>
<thead>
<tr>
<th>Layer ID</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_LAYER_ID_0_RD</td>
<td>VDC5_DISPSEL_BACK</td>
<td>Displays background color if not used.</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_1_RD</td>
<td>VDC5_DISPSEL_LOWER</td>
<td>Displays the lower layers if not used.</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_2_RD</td>
<td>VDC5_DISPSEL_LOWER</td>
<td>Displays the lower layers if not used.</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_3_RD</td>
<td>VDC5_DISPSEL_LOWER</td>
<td>Displays the lower layers if not used.</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_VIN_RD</td>
<td>VDC5_DISPSEL_LOWER</td>
<td>This function cannot change the graphics display mode for the VIN synthesizer. The graphics display mode for the VIN synthesizer is automatically set by the driver.</td>
</tr>
<tr>
<td>VDC5_LAYER_ID_OIR_RD</td>
<td>VDC5_DISPSEL_BACK</td>
<td>Displays background color if not used.</td>
</tr>
</tbody>
</table>

For the graphics display mode, VDC5_DISPSEL_BACK is specified to display background color, and VDC5_DISPSEL_CURRENT is specified to display the graphics. Other values are specified depending on the layer and its purpose of use.

### Table 2-12 Graphics Display Mode and Uses

<table>
<thead>
<tr>
<th>Layer ID</th>
<th>Value</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_LAYER_ID_0_RD</td>
<td>VDC5_DISPSEL_LOWER</td>
<td>• Displays the input video image.</td>
</tr>
<tr>
<td></td>
<td>VDC5_DISPSEL_BLEND</td>
<td>• Displays the enlarged graphics.</td>
</tr>
<tr>
<td></td>
<td>VDC5_LAYER_ID_1_RD</td>
<td>VDC5_DISPSEL_LOWER        • Displays the input video image.</td>
</tr>
<tr>
<td></td>
<td>VDC5_DISPSEL_BLEND</td>
<td>• Displays the enlarged graphics.</td>
</tr>
<tr>
<td></td>
<td>VDC5_LAYER_ID_2_RD / VDC5_LAYER_ID_3_RD</td>
<td>VDC5_DISPSEL_BLEND</td>
</tr>
<tr>
<td></td>
<td>VDC5_DISPSEL_BLEND</td>
<td>• Displays the blended image of lower-layer graphics and current graphics.</td>
</tr>
<tr>
<td></td>
<td>VDC5_LAYER_ID_OIR_RD</td>
<td>VDC5_DISPSEL_LOWER            • Setting prohibited.</td>
</tr>
<tr>
<td></td>
<td>VDC5_DISPSEL_BLEND</td>
<td>• Displays the graphics processed by the chroma-key.</td>
</tr>
</tbody>
</table>
2.3.12 R_VDC5_StopProcess

Synopsis
Data write/read stop processing

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_StopProcess(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id);

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_layer_id_t layer_id: Layer ID
  — VDC5_LAYER_ID_ALL: All enabled layers
  — VDC5_LAYER_ID_0_WR: Layer 0 write processing
  — VDC5_LAYER_ID_1_WR: Layer 1 write processing
  — VDC5_LAYER_ID_OIR_WR: OIR layer write processing
  — VDC5_LAYER_ID_0_RD: Layer 0 read processing
  — VDC5_LAYER_ID_1_RD: Layer 1 read processing
  — VDC5_LAYER_ID_2_RD: Layer 2 read processing
  — VDC5_LAYER_ID_3_RD: Layer 3 read processing
  — VDC5_LAYER_ID_OIR_RD: OIR layer read processing

Return value
• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  — VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

(1) Function

This function performs layer stop processing. If the layer ID specified in layer_id is VDC5_LAYER_ID_ALL, the function stops all the layers that are enabled and running. If the layer ID is not VDC5_LAYER_ID_ALL, the function stops only the specified layer.

When performing stop processing for write, the function stops the write to the frame buffer. When performing stop processing for read, the function stops the read from the frame buffer and resets the graphics display mode to the initial values for each of the layers. See Table 2-11 for the initial values of the graphics display mode.

(2) Use conditions

No particular use conditions are imposed if layer_id is found to be set to VDC5_LAYER_ID_ALL when the function is used. In the other cases, the conditions listed below apply. If these conditions are not met, the function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER).
• The specified layer is enabled.
• The specified layer is running.

Layers involved in write processing are enabled by calling the function R_VDC5_WriteDataControl and layers involved in read processing are enabled by calling the function R_VDC5_ReadDataControl. Layers that are in the stopped state can be started by calling the function R_VDC5_StartProcess.
### 2.3.13 R_VDC5_ReleaseDataControl

**Synopsis**
Data write/read control release processing

**Header**
r_vdc5.h

**Declaration**
```c
vdc5_error_t R_VDC5_ReleaseDataControl(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id);
```

**Arguments**
- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- `vdc5_layer_id_t layer_id`: Layer ID
  - VDC5_LAYER_ID_ALL: All enabled layers
  - VDC5_LAYER_ID_0_WR: Layer 0 write processing
  - VDC5_LAYER_ID_1_WR: Layer 1 write processing
  - VDC5_LAYER_ID_OIR_WR: OIR layer write processing
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
  - VDC5_LAYER_ID_OIR_RD: OIR layer read processing

**Return value**
- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

**Details**

1. **Function**
   This function performs the following processing:
   - Disables the specified layer.
   - Initializes the cascaded connection settings (layer 1 read processing only).
   - Sets up the color matrix in image quality improver 1 (layer 1 read processing only).
   - Initializes the VIN synthesizer (layer 0 read processing and layer 1 read processing only).

   If the layer ID specified in `layer_id` is VDC5_LAYER_ID_ALL, the function disables all the layers that are not running and also enabled. If the layer ID is not VDC5_LAYER_ID_ALL, the function disables only the specified layers.

2. **Use conditions**
   No particular use conditions are imposed if `layer_id` is found to be set to VDC5_LAYER_ID_ALL when the function is used. In the other cases, the conditions listed below apply. If these conditions are not met, the function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER).
   - The specified layer is enabled.
   - The specified layer is stopped.
Layers involved in write processing are enabled by calling the function `R_VDC5_WriteDataControl` and layers involved in read processing are enabled by calling the function `R_VDC5_ReadDataControl`. Layers that are running can be stopped by calling the function `R_VDC5_StopProcess`.
### R_VDC5_VideoNoiseReduction

**Synopsis**  
Noise reduction setup

**Header**  
r_vdc5.h

**Declaration**  
vdc5_error_t R_VDC5_VideoNoiseReduction(
    const vdc5_channel_t   ch,
    const vdc5_onoff_t   nr1d_on,
    const vdc5_noise_reduction_t  * const param);

**Arguments**  
- **vdc5_channel_t ch**: Channel  
  - VDC5_CHANNEL_0: Channel 0  
  - VDC5_CHANNEL_1: Channel 1  
- **vdc5_onoff_t nr1d_on**: Noise reduction ON/OFF setting  
- **vdc5_noise_reduction_t * param**: Noise reduction setup parameter  
  The setting is not changed if NULL is specified. If this parameter has never been set up after a hardware reset, the initial value that is defined in the hardware manual remains valid. See the description about the structure for the initial value.

**Return value**  
- **vdc5_error_t**: Error code  
  - VDC5_OK: Normal termination  
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error  
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error  
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error  
  - VDC5_ERR_RESOURCE_INPUT: Input signal resource error

**Details**

1. **Function**  
   This function performs the following noise reduction processing:
   - Turns on and off noise reduction processing.
   - Sets up the noise reduction parameters for the Y/G, Cb/B, and Cr/R signals.
   
   The setup of noise reduction parameters and noise reduction ON/OFF control can be made separately. Once set up noise reduction parameters remain valid until a hardware reset occurs or they are overwritten by this function with other settings.

2. **Use conditions**  
   Before this function is used, it is necessary to enable a video input by calling the function R_VDC5_VideoInput. If no video input is enabled, the function returns an input signal resource error (VDC5_ERR_RESOURCE_INPUT).

3. **Parameter details**  
   The members of the vdc5_noise_reduction_t are described below.
   typedef struct
   {
     vdc5_nr_param_t    y;
     vdc5_nr_param_t    cb;
     vdc5_nr_param_t    cr;
   } vdc5_noise_reduction_t;
<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_nr_param_t</td>
<td>nr1d_tap</td>
<td>0</td>
<td>TAP select</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_TAPSEL_1 (0): Adjacent pixel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_TAPSEL_2 (1): 2 adjacent pixels</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_TAPSEL_3 (2): 3 adjacent pixels</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_TAPSEL_4 (3): 4 adjacent pixels</td>
</tr>
<tr>
<td></td>
<td>nr1d_th</td>
<td>8</td>
<td>Maximum value of coring (absolute value)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0x0000 to 0x007F</td>
</tr>
<tr>
<td>vdc5_nr_gain_t</td>
<td>nr1d_gain</td>
<td>3</td>
<td>Noise reduction gain adjustment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_GAIN_1_2 (0): 1/2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_GAIN_1_4 (1): 1/4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_GAIN_1_8 (2): 1/8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_NR_GAIN_1_16 (3): 1/16</td>
</tr>
</tbody>
</table>
2.3.15  R_VDC5_ImageColorMatrix

Synopsis
Color matrix setup

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_ImageColorMatrix(
    const vdc5_channel_t ch,
    const vdc5_color_matrix_t * const param);

Arguments

- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_color_matrix_t * param: Color matrix setup parameter
  Do not specify NULL.

Return value

- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  - VDC5_ERR_PARAM_CONDITION: Unauthorized condition error
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

(1)  Function

This function sets up the specified color matrix. The VDC5 has three color matrixes for each channel (see Figure 2-10).

The color matrixes automatically set up by the VDC5 driver according to the color format to be used. Consequently, this function need not be used except when there is a need to change color matrix values dynamically. The automatic setup of the color matrixes by the VDC5 driver proceeds as follows:

- When the function R_VDC5_WriteDataControl is called for layer 0 write processing or layer 1 write processing
  - The driver determines the necessary color conversion from the input format of the video image and the frame buffer video-signal writing format that is designated by the function R_VDC5_WriteDataControl and sets it up in the color matrix in the input controller.
- When the function R_VDC5_ReadDataControl is called for layer 0 read processing
  - The driver determines the necessary color conversion from the format of the frame buffer read signal designated by the function R_VDC5_ReadDataControl and sets it up in the color matrix in image quality improver 0.
  - If the read process for layer 1 is not used, the driver sets up the parameters for GBR to GBR conversion for the color matrix in image quality improver 1.
- When the function R_VDC5_ReadDataControl is called for layer 1 read processing
  - The driver determines the necessary color conversion from the format of the frame buffer read signal designated by the function R_VDC5_ReadDataControl and sets it up in the color matrix in image quality improver 1.
- When the function R_VDC5_ReleaseDataControl is called for layer 1 read processing
  - The driver sets up the parameters for GBR to GBR conversion for the color matrix in image quality improver 1.

See 2.2.6 for the values of the color matrixes that are automatically set up by the VDC5 driver.
Figure 2-10  Color Matrixes

(2) **Use conditions**

When this function is to be used to make settings for the color matrix in the input controller, the write process for layer 0 in the channel designated by ch or the write process for layer 1 in another channel needs to be enabled. These layers are enabled by calling the function R_VDC5_WriteDataControl.

When this function is to be used to make settings for the color matrix in the image quality improver 0 or the image quality improver 1, the read processes for their layers 0 and 1 need to be enabled. These layers are enabled by calling the function R_VDC5_ReadDataControl.

The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if it is called when the layers associated with the color matrix are disabled.

(3) **Parameter details**

The members of the `vdc5_color_matrix_t` structure are described below.

```c
typedef struct
{
    vdc5_colormtx_module_t  module;
    vdc5_colormtx_mode_t    mtx_mode;
    uint16_t                offset[VDC5_COLORMTX_OFFST_NUM];
    uint16_t                gain[VDC5_COLORMTX_GAIN_NUM];
} vdc5_color_matrix_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_colormtx_module_t</td>
<td>module</td>
<td>Color matrix module</td>
</tr>
<tr>
<td>vdc5_colormtx_mode_t</td>
<td>mtx_mode</td>
<td>Operating mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_colormtx_module_t</td>
<td>Color matrix module</td>
</tr>
<tr>
<td>module</td>
<td>• VDC5_COLORMTX_IMGCNT (0): Input controller</td>
</tr>
<tr>
<td></td>
<td>• VDC5_COLORMTX_ADJ_0 (1): Image quality improver 0</td>
</tr>
<tr>
<td></td>
<td>• VDC5_COLORMTX_ADJ_1 (2): Image quality improver 1</td>
</tr>
<tr>
<td>vdc5_colormtx_mode_t</td>
<td>Operating mode</td>
</tr>
<tr>
<td>mtx_mode</td>
<td>• VDC5_COLORMTX_GBR_GBR: GBR → GBR</td>
</tr>
<tr>
<td></td>
<td>• VDC5_COLORMTX_GBR_YCBCR: GBR → YCbCr *</td>
</tr>
<tr>
<td></td>
<td>• VDC5_COLORMTX_YCBCR_GBR: YCbCr → GBR</td>
</tr>
<tr>
<td></td>
<td>• VDC5_COLORMTX_YCBCR_YCBCR: YCbCr → YCbCr *</td>
</tr>
</tbody>
</table>
uint16_t offset[VDC5_COLORMTX_OFFST_NUM]
Offset (DC) adjustment of Y/G, B, and R signal
0x0000 (-128) to 0x0080 (0) to 0x00FF (+127)

uint16_t gain[VDC5_COLORMTX_GAIN_NUM]
GG, GB, GR, BB, BR, RG, RB, and RR signal gain adjustment
Signed (2’s complement)
-1024 to +1023[LSB], 256[LSB] = 1.0[times]

Note: The operating mode in which conversion to YCbCr is performed is made available only when the input controller (VDC5_COLORMTX_IMGCNT) is specified in module.

vdc5_colormtx_offset_t is an enumeration type for representing the color matrix offset.

typedef enum
{
    VDC5_COLORMTX_OFFST_YG = 0,
    VDC5_COLORMTX_OFFST_B,
    VDC5_COLORMTX_OFFST_R,
    VDC5_COLORMTX_OFFST_NUM
} vdc5_colormtx_offset_t;

<table>
<thead>
<tr>
<th>Enumeration Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_COLORMTX_OFFST_YG</td>
<td>0</td>
<td>Offset (DC) adjustment of Y/G signal</td>
</tr>
<tr>
<td>VDC5_COLORMTX_OFFST_B</td>
<td>1</td>
<td>Offset (DC) adjustment of B signal</td>
</tr>
<tr>
<td>VDC5_COLORMTX_OFFST_R</td>
<td>2</td>
<td>Offset (DC) adjustment of R signal</td>
</tr>
<tr>
<td>VDC5_COLORMTX_OFFST_NUM</td>
<td>3</td>
<td>Number of color matrix offset parameters</td>
</tr>
</tbody>
</table>

vdc5_colormtx_gain_t is an enumeration type for representing the color matrix gain.

typedef enum
{
    VDC5_COLORMTX_GAIN_GG = 0,
    VDC5_COLORMTX_GAIN_GB,
    VDC5_COLORMTX_GAIN_GR,
    VDC5_COLORMTX_GAIN_BG,
    VDC5_COLORMTX_GAIN_BG,
    VDC5_COLORMTX_GAIN_GR,
    VDC5_COLORMTX_GAIN_BG,
    VDC5_COLORMTX_GAIN_BB,
    VDC5_COLORMTX_GAIN_RG,
    VDC5_COLORMTX_GAIN_RB,
    VDC5_COLORMTX_GAIN_RR,
    VDC5_COLORMTX_GAIN_NUM
} vdc5_colormtx_gain_t;

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_COLORMTX_GAIN_GG</td>
<td>0</td>
<td>Y/G signal gain adjustment for Y/G signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_GB</td>
<td>1</td>
<td>Cb/B signal gain adjustment for Y/G signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_GR</td>
<td>2</td>
<td>Cr/R signal gain adjustment for Y/G signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_BG</td>
<td>3</td>
<td>Y/G signal gain adjustment for Cb/B signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_BG</td>
<td>4</td>
<td>Cb/B signal gain adjustment for Cb/B signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_BR</td>
<td>5</td>
<td>Cr/R signal gain adjustment for Cb/B signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_RG</td>
<td>6</td>
<td>Y/G signal gain adjustment for Cr/R signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_RB</td>
<td>7</td>
<td>Cb/B signal gain adjustment for Cr/R signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_RR</td>
<td>8</td>
<td>Cr/R signal gain adjustment for Cr/R signal output</td>
</tr>
<tr>
<td>VDC5_COLORMTX_GAIN_NUM</td>
<td>9</td>
<td>Number of color matrix gain parameters</td>
</tr>
</tbody>
</table>
### 2.3.16 R_VDC5_ImageEnhancement

#### Synopsis
Image enhancement processing

#### Header
r_vdc5.h

#### Declaration
```c
vdc5_error_t R_VDC5_ImageEnhancement(
    const vdc5_channel_t   ch,
    const vdc5_imgimprv_id_t  imgimprv_id,
    const vdc5_onoff_t   shp_h_on,
    const vdc5_enhance_sharp_t  * const sharp_param,
    const vdc5_onoff_t   lti_h_on,
    const vdc5_enhance_lti_t  * const lti_param,
    const vdc5_period_rect_t  * const enh_area);
```

#### Arguments
- `vdc5_channel_t ch`: Channel
  - `VDC5_CHANNEL_0`: Channel 0
  - `VDC5_CHANNEL_1`: Channel 1
- `vdc5_imgimprv_id_t imgimprv_id`: Image quality improver ID
  - `VDC5_IMG_IMPRV_0`: Image quality improver 0
  - `VDC5_IMG_IMPRV_1`: Image quality improver 1
- `vdc5_onoff_t shp_h_on`: Sharpness ON/OFF setting
- `vdc5_enhance_sharp_t * sharp_param`: Sharpness setup parameter
  The setting is not changed if NULL is specified.
- `vdc5_onoff_t lti_h_on`: LTI ON/OFF setting
- `vdc5_enhance_lti_t * lti_param`: LTI setup parameter
  The setting is not changed if NULL is specified.
- `vdc5_period_rect_t * enh_area`: Enhancer-enabled area setup parameter
  The setting is not changed if NULL is specified.

If parameters described above have never been set up after a hardware reset, the initial values that are defined in the hardware manual remain valid. See the description about the structure for the initial value.

#### Return value
- `vdc5_error_t`: Error code
  - `VDC5_OK`: Normal termination
  - `VDC5_ERR_PARAM_CHANNEL`: Channel invalid error
  - `VDC5_ERR_PARAM_BIT_WIDTH`: Bit width error
  - `VDC5_ERR_PARAM_UNDEFINED`: Undefined parameter specification error
  - `VDC5_ERR_PARAM_EXCEED_RANGE`: Out-of-value-range error
  - `VDC5_ERR_IF_CONDITION`: Interface condition error
  - `VDC5_ERR_RESOURCE_LAYER`: Layer resource error

#### Details

(1) **Function**

This function performs the following image quality improvement processing:

- Turns on and off sharpness processing.
- Sets up the sharpness parameter.
- Turns on and off LTI processing.
- Sets up the LTI parameter.
- Sets up the enhancer-enabled area to be subjected to sharpness and LTI processing.
The setup of parameters and ON/OFF control for sharpness and LTI processing can be made separately. The parameters that are once set up remain valid until a hardware reset occurs or they are overwritten by this function with other settings.

(2) **Use conditions**

When this function is to be used, the layer associated with the image quality improver specified in imgimprv_id needs to be enabled. Image quality improver 0 is associated with the read process for layer 0 and image quality improver 1 with the read process for layer 1. The layers are enabled by calling the function R_VDC5_ReadDataControl for each of the layers. The function returns a layer resource error (VDC5_ERRRESOURCE_LAYER) if the pertinent layer is disabled.

This processing is inhibited if the color format (format of the frame buffer read signal) of the layer associated with the specified image quality improver is set to the format other than YCbCr422 and YCbCr444. In such a case, the function returns an interface condition error (VDC5_ERRINTERFACE_CONDITION).

(3) **Parameter details**

vdc5_enhance_sharp_t structure are described below.

```c
typedef struct {
    vdc5_onoff_t            shp_h2_lpf_sel;
    vdc5_sharpness_ctrl_t   hrz_sharp[VDC5_IMGENH_SHARP_NUM];
} vdc5_enhance_sharp_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_onoff_t</td>
<td>shp_h2_lpf_sel</td>
<td>VDC5_OFF (0)</td>
<td>LPF selection for folding prevention before H2 edge detection</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_OFF: LPF not selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_ON: LPF selected</td>
</tr>
<tr>
<td>vdc5_sharpness_ctrl_t</td>
<td>hrz_sharp [VDC5_IMGENH_SHARP_NUM]</td>
<td>-</td>
<td>Sharpness control parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Horizontal sharpness (H1, H2, H3)</td>
</tr>
</tbody>
</table>

vdc5_img_enh_sh_t is an enumeration type for representing the sharpness band.

```c
typedef enum {
    VDC5_IMGENH_SHARP_H1 = 0,
    VDC5_IMGENH_SHARP_H2,
    VDC5_IMGENH_SHARP_H3,
    VDC5_IMGENH_SHARP_NUM
} vdc5_img_enh_sh_t;
```

<table>
<thead>
<tr>
<th>Enumeration constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5_IMGENH_SHARP_H1</td>
<td>0</td>
<td>Horizontal sharpness (H1)</td>
</tr>
<tr>
<td>VDC5_IMGENH_SHARP_H2</td>
<td>1</td>
<td>Horizontal sharpness (H2)</td>
</tr>
<tr>
<td>VDC5_IMGENH_SHARP_H3</td>
<td>2</td>
<td>Horizontal sharpness (H3)</td>
</tr>
<tr>
<td>VDC5_IMGENH_SHARP_NUM</td>
<td>3</td>
<td>Number of horizontal sharpness bands</td>
</tr>
</tbody>
</table>

The members of the vdc5_sharpness_ctrl_t structure are described below.

```c
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; } vdc5_sharpness_ctrl_t;

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>shp_clip_o</td>
<td>0</td>
<td>Sharpness correction value clipping (on the overshoot side)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 to 0x00FF</td>
</tr>
<tr>
<td>uint8_t</td>
<td>shp_clip_u</td>
<td>0</td>
<td>Sharpness correction value clipping (on the undershoot side)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 to 0x00FF</td>
</tr>
<tr>
<td>uint8_t</td>
<td>shp_gain_o</td>
<td>0</td>
<td>Sharpness edge amplitude value gain (on the overshoot side)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 (0 time) to 0x0040 (1 time) to 0x00FF (approx. 4 times)</td>
</tr>
<tr>
<td>uint8_t</td>
<td>shp_gain_u</td>
<td>0</td>
<td>Sharpness edge amplitude value gain (on the undershoot side)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 (0 time) to 0x0040 (1 time) to 0x00FF (approx. 4 times)</td>
</tr>
<tr>
<td>uint8_t</td>
<td>shp_core</td>
<td>0</td>
<td>Active sharpness range</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 to 0x007F</td>
</tr>
</tbody>
</table>

The members of the vdc5_enhance_lti_t structure are described below.

typedef struct
{    vdc5_onoff_t            lti_h2_lpf_sel;
    vdc5_lti_mdfil_sel_t    lti_h4_median_tap_sel;
    vdc5_lti_ctrl_t         lti[VDC5_IMGENH_LTI_NUM];
} vdc5_enhance_lti_t;

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_onoff_t</td>
<td>lti_h2_lpf_sel</td>
<td>VDC5_OFF (0)</td>
<td>LPF selection for folding prevention before H2 edge detection</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_OFF: LPF not selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_ON: LPF selected</td>
</tr>
<tr>
<td>vdc5_lti_mdfil_sel_t</td>
<td>lti_h4_median_tap_sel</td>
<td>0</td>
<td>Median filter reference pixel select</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_LTI_MDFIL_SEL_ADJ2 (0):</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Second adjacent pixel selected as reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_LTI_MDFIL_SEL_ADJ1 (1):</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Second adjacent pixel selected as reference</td>
</tr>
<tr>
<td>vdc5_lti_ctrl_t</td>
<td>lti[VDC5_IMGENH_LTI_NUM]</td>
<td>-</td>
<td>LTI control parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Horizontal LTI (H2, H4)</td>
</tr>
</tbody>
</table>

vdc5_img_enh_lti_t is an enumeration type for representing the LTI band.

typedef enum
{    VDC5_IMGENH_LTI1 = 0,
    VDC5_IMGENH_LTI2,
    VDC5_IMGENH_LTI_NUM
}
The members of the `vdc5_lti_ctrl_t` structure are described below.

```c
typedef struct {
    uint8_t lti_inc_zero;
    uint8_t lti_gain;
    uint8_t lti_core;
} vdc5_lti_ctrl_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>lti_inc_zero</td>
<td>10</td>
<td>Median filter LTI correction threshold</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 to 0x00FF</td>
</tr>
<tr>
<td></td>
<td>lti_gain</td>
<td>0</td>
<td>LTI edge amplitude value gain</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 (0 time) to 0x0040 (1 time) to 0x00FF (approx. 4 times)</td>
</tr>
<tr>
<td></td>
<td>lti_core</td>
<td>0</td>
<td>LTI coring</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0x0000 to 0x00FF</td>
</tr>
</tbody>
</table>

See also 2.2.3(1) for the `vdc5_period_rect_t` structure.

```c
uint16_t vs;
uint16_t vw;
uint16_t hs;
uint16_t hw;
```
2.3.17 R_VDC5_ImageBlackStretch

Synopsis
Black stretch setup

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_ImageBlackStretch(
    const vdc5_channel_t ch,
    const vdc5_imgimprv_id_t imgimprv_id,
    const vdc5_onoff_t bkstr_on,
    const vdc5_black_t * const param);

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_imgimprv_id_t imgimprv_id: Image quality improver ID
  — VDC5_IMG_IMPRV_0: Image quality improver 0
  — VDC5_IMG_IMPRV_1: Image quality improver 1
• vdc5_onoff_t bkstr_on: Black stretch ON/OFF setting
• vdc5_black_t * param: Black stretch setup parameter
  The setting is not changed if NULL is specified. If this parameter has never been set up after a hardware reset, the initial value that is defined in the hardware manual remains valid. See the description about the structure for the initial value.

Return value
• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  — VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  — VDC5_ERR_IF_CONDITION: Interface condition error
  — VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details
(1) Function
This function performs the following black stretch processing for the specified image quality improver:
• Turns on and off black stretch processing.
• Sets up the black stretch parameters.

The setup of parameters and ON/OFF control of the black stretch processing can be made separately. The settings once established by this function remain valid until a hardware reset occurs or they are overwritten by this function with other settings.

(2) Use conditions
When this function is to be used, the layer associated with the image quality improver specified in imgimprv_id needs to be enabled. Image quality improver 0 is associated with the read process for layer 0 and image quality improver 1 with the read process for layer 1. The layers are enabled by calling the function R_VDC5_ReadDataControl. The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the pertinent layer is disabled.
The execution of this processing is inhibited if the color format (format of the frame buffer read signals) of the layer associated with the specified image quality improver is set to a format other than YCbCr422 and YCbCr444. In such a case, the function returns an interface condition error (VDC5_ERR_IF_CONDITION).

(3) Parameter details

vdc5_black_t structure are described below.

typedef struct
{
    uint16_t    bkstr_st;
    uint16_t    bkstr_d;
    uint16_t    bkstr_t1;
    uint16_t    bkstr_t2;
} vdc5_black_t;

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>bkstr_st</td>
<td>0</td>
<td>Black stretch start point</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 (low) to 15 (high)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>bkstr_d</td>
<td>0</td>
<td>Black stretch depth</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 (shallow) to 15 (deep)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>bkstr_t1</td>
<td>0</td>
<td>Black stretch time constant (T1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 (small) to 31 (large)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>bkstr_t2</td>
<td>0</td>
<td>Black stretch time constant (T2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 (small) to 30 (large)</td>
</tr>
</tbody>
</table>
2.3.18  R_VDC5_AlphaBlending

Synopsis  Alpha blending setup

Header  r_vdc5.h

Declaration  vdc5_error_t R_VDC5_AlphaBlending(
              const vdc5_channel_t   ch,
              const vdc5_layer_id_t   layer_id,
              const vdc5_alpha_blending_t  * const param);

Arguments
  •  vdc5_channel_t ch: Channel
    —  VDC5_CHANNEL_0: Channel 0
    —  VDC5_CHANNEL_1: Channel 1
  •  vdc5_layer_id_t layer_id: Layer ID
    —  VDC5_LAYER_ID_1_RD: Layer 1 read processing
    —  VDC5_LAYER_ID_2_RD: Layer 2 read processing
    —  VDC5_LAYER_ID_3_RD: Layer 3 read processing
  •  vdc5_alpha_blending_t * param: Alpha blending setup parameter
    Do not specify NULL.

Return value
  •  vdc5_error_t: Error code
    —  VDC5_OK: Normal termination
    —  VDC5_ERR_PARAM_CHANNEL: Channel invalid error
    —  VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
    —  VDC5_ERR_PARAM_NULL: NULL specification error
    —  VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

(1)  Function

This function performs the following processing for alpha blending except for rectangle alpha blending:

•  Sets up the alpha value of the ARGB1555/RGBA5551 formats.
•  Make settings for premultiplication processing at alpha blending in one-pixel.

This function can set up the alpha value of ARGB1555/RGBA5551 for layer1 to 3 read processes. The alpha value ARGB1555/RGBA5551 which is used for layer 0 and output image generator read processes is automatically set to '255 (= 1.0, nontransparent)' by the driver.

(2)  Use conditions

When this function is to be used, the layer specified in layer_id needs to be enabled. The layer is enabled by calling and executing the function R_VDC5_ReadDataControl on that layer. The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the layer specified in layer_id is found disabled.

(3)  Parameter details

The members of the vdc5_alpha_blending_t structure are described below.

typedef struct
{
    const vdc5_alpha_argb1555_t   * alpha_1bit;
    const vdc5_alpha_pixel_t      * alpha_pixel;
}
### vdc5_alpha_blending_t

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>const vdc5_alpha_argb1555_t*</td>
<td>alpha_1bit</td>
<td>Alpha signal of the ARGB1555/RGBA5551 formats</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The setting is not changed if NULL is specified.</td>
</tr>
<tr>
<td>const vdc5_alpha_pixel_t*</td>
<td>alpha_pixel</td>
<td>Premultiplication processing at alpha blending in one-pixel</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The setting is not changed if NULL is specified.</td>
</tr>
</tbody>
</table>

If the parameters alpha_1bit and alpha_pixel have never been set up after a hardware reset, their initial values that are defined in the hardware manual remain valid. See the description about the structures for the initial values.

The members of the `vdc5_alpha_argb1555_t` structure are described below.

```c
typedef struct {
  uint8_t  gr_a0;
  uint8_t  gr_a1;
} vdc5_alpha_argb1555_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>gr_a0</td>
<td>0</td>
<td>Alpha signal when alpha is set to '0'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 255</td>
<td></td>
</tr>
<tr>
<td>uint8_t</td>
<td>gr_a1</td>
<td>0</td>
<td>Alpha signal when alpha is set to '1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 255</td>
<td></td>
</tr>
</tbody>
</table>

The members of the `vdc5_alpha_pixel_t` structure are described below.

```c
typedef struct {
  vdc5_onoff_t  gr_acalc_md;
} vdc5_alpha_pixel_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_onoff_t</td>
<td>gr_acalc_md</td>
<td>VDC5_OFF (0)</td>
<td>Premultiplication processing at alpha blending in one-pixel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>units</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_ON</td>
</tr>
</tbody>
</table>
### 2.3.19 R_VDC5_AlphaBlendingRect

**Synopsis**
Rectangle alpha blending setup

**Header**
r_vdc5.h

**Declaration**
```c
vdc5_error_t R_VDC5_AlphaBlendingRect(
    const vdc5_channel_t   ch,
    const vdc5_layer_id_t   layer_id,
    const vdc5_onoff_t   gr_arc_on,
    const vdc5_alpha_blending_rect_t * const param);
```

**Arguments**
- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- `vdc5_layer_id_t layer_id`: Layer ID
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
  - VDC5_LAYER_ID_VIN_RD: VIN synthesizer
- `vdc5_onoff_t gr_arc_on`: ON/OFF setting for alpha blending in a rectangular area
- `vdc5_alpha_blending_rect_t * param`: Setup parameter for alpha blending in a rectangular area
  The setting is not changed if NULL is specified. See the description about the structure for the initial value.

**Return value**
- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  - VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  - VDC5_ERR_IF_CONDITION: Interface condition error
  - VDC5_ERRRESOURCE_LAYER: Layer resource error

**Details**

1. **Function**
   - This function performs the following processing for rectangle alpha blending:
     - Turns on and off alpha blending in a rectangular area.
     - Sets up the rectangular area subjected to alpha blending.
     - Sets up the alpha value for alpha blending in a rectangular area.
     - Makes fade-in/-out settings to be applied to rectangle alpha blending.
     - Allocates graphics 0 and 1 to the lower-layer/current graphics in the VIN synthesizer.

   The setup and ON/OFF control of the alpha blending in rectangular area can be made separately. The alpha blending settings once established by this function remain valid until a hardware reset occurs, until overwritten with other settings, or until the specified layer resources are destroyed by the function R_VDC5_ReleaseDataControl.

   The VIN synthesizer can be made available for synthesizing two input video images by specifying VDC5_LAYER_ID_VIN_RD in `layer_id`. 
(2) Use conditions

When this function is to be used specifying a layer other than VIN synthesizer in layer_id, the specified layer needs to be enabled. The layer is enabled by calling the function R_VDC5_ReadDataControl on that layer. The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the layer specified in layer_id is found disabled.

The execution of alpha blending processing is inhibited if layer_id is set to a layer other than VIN synthesizer and the color format (format of the frame buffer read signals) of that layer is set to YCbCr422 or YCbCr444. In such a case, the function returns an interface condition error (VDC5_ERR_IF_CONDITION).

(3) Parameter details

The members of the vdc5_alpha_blending_rect_t structure are described below.

```c
typedef struct
{
    const vdc5_pd_disp_rect_t * gr_arc;
    const vdc5_alpha_rect_t   * alpha_rect;
    const vdc5_scl_und_sel_t  * scl_und_sel;
} vdc5_alpha_blending_rect_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_pd_disp_rect_t *</td>
<td>gr_arc</td>
<td>Rectangular area subjected to alpha blending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The setting is not changed if NULL is specified.</td>
</tr>
<tr>
<td>vdc5_alpha_rect_t *</td>
<td>alpha_rect</td>
<td>Parameter for alpha blending in a rectangular area</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The setting is not changed if NULL is specified.</td>
</tr>
<tr>
<td>vdc5_scl_und_sel_t *</td>
<td>scl_und_sel</td>
<td>Selection of lower-layer plane in scaler</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The setting is not changed if NULL is specified.</td>
</tr>
</tbody>
</table>

The parameter gr_arc is initialized by the driver to the same graphics display area when the API function R_VDC5_ReadDataControl is called. The parameters alpha_rect and scl_und_sel are kept at their initial values that are defined in the hardware manual if they have never been set up after a hardware reset. See the structure descriptions given below for their initial values. The parameters in vdc5_alpha_blending_rect_t structure, including the parameters gr_arc, alpha_rect, and scl_und_sel are kept at their initial values if they have never been set up after a hardware reset.

When two input video images are synthesized by the VIN synthesizer, the VDC5 driver initializes the rectangle area to be subjected to rectangle alpha blending with the graphics display area of the upper layer during the execution of the function R_VDC5_ReadDataControl. Unless the setting of the lower layer plane of the scaler is changed during this function, scaler 1 (graphics 1) remains to be the upper layer.

```c
typedef struct
{
    uint16_t   vs_rel;
    uint16_t   vw_rel;
    uint16_t   hs_rel;
    uint16_t   hw_rel;
} vdc5_pd_disp_rect_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vs_rel</td>
<td>0</td>
<td>Vertical start position of the valid image area for alpha blending in a rectangular area (lines)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This is the relative position from the vertical start position of the graphics display area.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>uint16_t vw_rel</td>
<td>Vertical width of the valid image area for alpha blending in a rectangular area (lines)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>uint16_t hs_rel</td>
<td>Horizontal start position of the valid image area for alpha blending in a rectangular area (clock cycles). This is the relative position from the horizontal start position of the graphics display area.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>uint16_t hw_rel</td>
<td>Horizontal width of the valid image area for alpha blending in a rectangular area (clock cycles)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The effective image area vertical width and horizontal width are initialized by the driver to the same values of the graphics display area.

The rectangular area to be subjected to alpha blending is specified as a relative position within the graphics display area for the pertinent layer that is specified through the function R_VDC5_ReadDataControl (see Figure 2-11).

![Figure 2-11 Rectangular Area Setting for Alpha Blending](image)

Area A: Graphics display area
See 2.3.9(3) and 2.2.3(1) for details.
Area B: Rectangular area for alpha blending

Even when the graphics display area is changed by the function R_VDC5_ChangeReadProcess, the rectangular area for alpha blending does not follow the change. To change the graphics display area when using alpha blending in a rectangular area, it is also necessary to change the rectangular area for alpha blending.

The members of the vdc5_alpha_rect_t structure are described below.

```c
typedef struct {
    int16_t gr_arc_coef;
    uint8_t gr_arc_rate;
} vdc5_alpha_rect_t;
```
```c
uint8_t gr_arc_def;
vdc5_onoff_t gr_arc_mul;
} vdc5_alpha_rect_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| int16_t  | gr_arc_coef     | 0             | Alpha coefficient for alpha blending in a rectangular area
|          |                 |               | Variation (-255 to +255) |
| uint8_t  | gr_arc_rate     | 0             | Frame rate for alpha blending in a rectangular area
|          |                 |               | gr_arc_coef is added to the alpha value every time Vsync rises the number of times equal to gr_arc_rate + 1. |
|          |                 |               | 0 to 255   |
| uint8_t  | gr_arc_def      | 255           | Initial alpha value for alpha blending in a rectangular area |
|          |                 |               | 0 to 255   |
| vdc5_onoff_t | gr_arc_mul     | VDC5_OFF (0) | Multiplication processing with current alpha at alpha blending in a rectangular area |
|           |                 |               | • VDC5_OFF |
|           |                 |               | • VDC5_ON  |

The members of the vdc5_scl_und_sel_t structure are described below.

```c
typedef struct
{
    vdc5_onoff_t gr_vin_scl_und_sel;
} vdc5_scl_und_sel_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_onoff_t</td>
<td>gr_vin_scl_und_sel</td>
<td>VDC5_OFF (0)</td>
<td>Selection of lower-layer plane in scaler</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_OFF:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Selects graphics 0 as lower-layer graphics and graphics 1 as current graphics</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• VDC5_ON:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Selects graphics 1 as lower-layer graphics and graphics 0 as current graphics</td>
</tr>
</tbody>
</table>

The gr_vin_scl_und_sel setting is referenced when synthesizing two input video images using two scalers. This setting has no effect in the other cases.
2.3.20 R_VDC5_Chromakey

Synopsis
Chroma-key setup

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_Chromakey(
    const vdc5_channel_t ch,
    const vdc5_layer_id_t layer_id,
    const vdc5_onoff_t gr_ck_on,
    const vdc5_chromakey_t * const param);

Arguments

• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_layer_id_t layer_id: Layer ID
  — VDC5_LAYER_ID_0_RD: Layer 0 read processing
  — VDC5_LAYER_ID_1_RD: Layer 1 read processing
  — VDC5_LAYER_ID_2_RD: Layer 2 read processing
  — VDC5_LAYER_ID_3_RD: Layer 3 read processing
  — VDC5_LAYER_ID_OIR_RD: OIR layer read processing
• vdc5_onoff_t gr_ck_on: Chroma-key ON/OFF setting
• vdc5_chromakey_t * param: Chroma-key setup parameter
  The setting is not changed if NULL is specified. If this parameter has never been set up after a hardware reset, the initial value that is defined in the hardware manual remains valid. See the description about the structure for the initial value.

Return value

• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_IF_CONDITION: Interface condition error
  — VDC5_ERR_RESOURCE_LAYER: Layer resource error

Details

(1) Function

This function performs the following chroma-key related processing:

• Turns on and off the chroma-key processing.
• Sets up the color signals to be subject to chroma-key processing and the color signals after replacement.

The setup of chroma-key processing and its ON/OFF control can be made separately. Once set up chroma-key settings remain valid until a hardware reset occurs, until overwritten with other settings, or until the specified layer resources are destroyed by the function R_VDC5_ReleaseDataControl.

(2) Use conditions

When this function is to be used, the layer specified in layer_id needs to be enabled. The layer is enabled by calling the function R_VDC5_ReadDataControl on that layer. The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the layer specified in layer_id is found disabled.
The execution of chroma-key processing is inhibited if the color format (format of the frame buffer read signals) of the specified layer is set to YCbCr422 or YCbCr444. In such a case, the function returns an interface condition error (VDC5_ERR_IF_CONDITION).

(3) Parameter details

vdc5_chromakey_t structure are described below.

```c
typedef struct {
    uint32_t    ck_color;
    uint32_t    rep_color;
    uint8_t     rep_alpha;
} vdc5_chromakey_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint32_t</td>
<td>ck_color</td>
<td>0</td>
<td>RGB/CLUT signal for RGB-index/CLUT-index chroma-key processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specify in the color format that is used in the target layer (LSB justified).</td>
</tr>
<tr>
<td>uint32_t</td>
<td>rep_color</td>
<td>0</td>
<td>Replaced RGB signal after RGB/CLUT-index chroma-key processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specify in the color format that is used in the target layer (LSB justified).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specify, however, in the RGB888 format if the color format is set to CLUT8,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CLUT4, or CLUT1. The alpha value in this parameter is ignored. Specify the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>replaced alpha signal in rep_alpha.</td>
</tr>
<tr>
<td>uint8_t</td>
<td>rep_alpha</td>
<td>0 *</td>
<td>Replaced alpha signal after RGB/CLUT-index chroma-key processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specify an alpha value in 8 bits.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 ~ 255</td>
</tr>
</tbody>
</table>

Note: The alpha value for layer 0 and output image generator is automatically set by the driver to '255'.
### 2.3.21 R_VDC5_CLUT

**Synopsis**
CLUT setup

**Header**
r_vdc5.h

**Declaration**
```c
vdc5_error_t R_VDC5_CLUT(
    const vdc5_channel_t ch,
    const vdc5_layer_id_t layer_id,
    const vdc5_clut_t * const param);
```

**Arguments**
- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- `vdc5_layer_id_t layer_id`: Layer ID
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
  - VDC5_LAYER_ID_OIR_RD: OIR layer read processing
- `vdc5_clut_t * param`: CLUT setup parameter
  Do not specify NULL.

**Return value**
- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_LAYER_ID: Invalid layer ID error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_EXCEED_RANGE: Out-of-value-range error
  - VDC5_ERR_RESOURCE_LAYER: Layer resource error

**Details**

(1) **Function**
This function sets up CLUT for the specified layer.

(2) **Use conditions**
When this function is to be used, the layer specified in `layer_id` needs to be enabled. The layer is enabled by calling and executing the function `R_VDC5_ReadDataControl`. The function returns a layer resource error (VDC5_ERR_RESOURCE_LAYER) if the layer specified in `layer_id` is found disabled.

(3) **Parameter details**
The members of the `vdc5_clut_t` structure are described below.
```c
typedef struct {
    uint32_t            color_num;
    const uint32_t    * clut;
} vdc5_clut_t;
```
<table>
<thead>
<tr>
<th>Type Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>ASCII string indicating pixel depth of frame buffer (0: 8-bit, 1: 16-bit)</td>
</tr>
<tr>
<td>uint32_t</td>
<td>Number of colors in CLUT</td>
</tr>
<tr>
<td>color_num</td>
<td>When CLUT1 format is used: 1 to 2</td>
</tr>
<tr>
<td></td>
<td>When CLUT4 format is used: 1 to 16</td>
</tr>
<tr>
<td></td>
<td>When CLUT8 format is used: 1 to 256</td>
</tr>
<tr>
<td>const uint32_t *</td>
<td>Address of the area storing the CLUT data (in ARGB8888 format)</td>
</tr>
<tr>
<td>clut</td>
<td>Do not specify NULL.</td>
</tr>
</tbody>
</table>
2.3.22  R_VDC5_DisplayCalibration

Synopsis
Display calibration processing

Header
r_vdc5.h

Declaration
vdc5_error_t R_VDC5_DisplayCalibration(
    const vdc5_channel_t   ch,
    const vdc5_disp_calibration_t  * const param);

Arguments
- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_disp_calibration_t * param: Display calibration parameter
  Do not specify NULL.

Return value
- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  - VDC5_ERR_PARAM_NULL: NULL specification error
  - VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  - VDC5_ERR_PARAM_UNDEFINED: Undefined parameter specification error
  - VDC5_ERR_RESOURCE_OUTPUT: Output resource error

Details
(1)  Function
This function performs the following processing for display calibration:
- Sets up panel brightness adjustment.
- Sets up contrast adjustment.
- Sets up panel dithering.
- Makes control settings for the correction circuit sequence.

The settings established by this function remain valid until a hardware reset occurs or they are overwritten by this function with other settings.

(2)  Use conditions
Before using this function, it is necessary to set up display output by calling the function R_VDC5_DisplayOutput. The function returns an output resource error (VDC5_ERRRESOURCE_OUTPUT) if the display output is not set up.

(3)  Parameter details
The members of the vdc5_disp_calibration_t structure are described below.

typedef struct
{
    vdc5_calibr_route_t             route;
    const vdc5_calibr_bright_t    * bright;
    const vdc5_calibr_contrast_t  * contrast;
    const vdc5_calibr_dither_t    * panel_dither;
} vdc5_disp_calibration_t;
### vdc5_calibr_route_t

**Type**
- route

**Member Name**
- vdc5_calibr_route_t

**Description**
- Correction circuit sequence control
  - VDC5_CALIBR_ROUTE_BCG: Brightness \(\rightarrow\) contrast \(\rightarrow\) gamma correction
  - VDC5_CALIBR_ROUTE_GBC: Gamma correction \(\rightarrow\) brightness \(\rightarrow\) contrast

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>const vdc5_calibr_bright_t *</td>
<td>bright</td>
<td></td>
<td>Brightness (DC) adjustment parameter Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const vdc5_calibr_contrast_t *</td>
<td>contrast</td>
<td></td>
<td>Contrast (gain) adjustment parameter Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const vdc5_calibr_dither_t *</td>
<td>panel_dither</td>
<td></td>
<td>Panel dithering parameter Specify NULL when this parameter value need not be changed.</td>
</tr>
</tbody>
</table>

If the brightness, contrast, and panel_dither parameters have never been set up after a hardware reset, their initial values that are defined in the hardware manual remain valid. See the structure descriptions given below for their initial values.

The members of the **vdc5_calibr_bright_t** structure are described below.

```c
typedef struct {
    uint16_t      pbtr_g;
    uint16_t      pbtr_b;
    uint16_t      pbtr_r;
} vdc5_calibr_bright_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint16_t</td>
<td>pbtr_g</td>
<td>512</td>
<td>Brightness (DC) adjustment of G signal 0x0000 (-512) to 0x03FF (+511)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>pbtr_b</td>
<td>512</td>
<td>Brightness (DC) adjustment of B signal 0x0000 (-512) to 0x03FF (+511)</td>
</tr>
<tr>
<td>uint16_t</td>
<td>pbtr_r</td>
<td>512</td>
<td>Brightness (DC) adjustment of R signal 0x0000 (-512) to 0x03FF (+511)</td>
</tr>
</tbody>
</table>

The members of the **vdc5_calibr_contrast_t** structure are described below.

```c
typedef struct {
    uint8_t       cont_g;
    uint8_t       cont_b;
    uint8_t       cont_r;
} vdc5_calibr_contrast_t;
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>cont_g</td>
<td>128</td>
<td>Contrast (gain) adjustment of G signal 0x0000 (0/128[times]) to 0x00FF (255/128[times])</td>
</tr>
<tr>
<td>uint8_t</td>
<td>cont_b</td>
<td>128</td>
<td>Contrast (gain) adjustment of B signal</td>
</tr>
</tbody>
</table>
cont_b | 0x0000 (0/128[times]) to 0x00FF (255/128[times])
---|---
uint8_t | 128 Contrast (gain) adjustment of R signal
cont_r | 0x0000 (0/128[times]) to 0x00FF (255/128[times])

The members of the vdc5_calibr_dither_t structure are described below.

```c
typedef struct
{
    vdc5_panel_dither_md_t  pdth_sel;
    uint8_t                 pdth_pa;
    uint8_t                 pdth_pb;
    uint8_t                 pdth_pc;
    uint8_t                 pdth_pd;
} vdc5_calibr_dither_t;
```

<table>
<thead>
<tr>
<th>Type Member Name</th>
<th>Initial Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdc5_panel_dither_md_t</td>
<td>0</td>
<td>Panel dither operation mode</td>
</tr>
<tr>
<td>pdth_sel</td>
<td>0</td>
<td>VDC5_PDTH_MD_TRU (0): Truncate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VDC5_PDTH_MD_RDOF (1): Round-off</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VDC5_PDTH_MD_2X2 (2): 2x2 pattern dither</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VDC5_PDTH_MD_RAND (3): Random pattern dither</td>
</tr>
<tr>
<td>uint8_t pdth_pa</td>
<td>3</td>
<td>Pattern value (A) of 2x2 pattern dither</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced only when pdth_sel is set to VDC5_PDTH_MD_2X2.</td>
</tr>
<tr>
<td>uint8_t pdth_pb</td>
<td>0</td>
<td>Pattern value (B) of 2x2 pattern dither</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced only when pdth_sel is set to VDC5_PDTH_MD_2X2.</td>
</tr>
<tr>
<td>uint8_t pdth_pc</td>
<td>2</td>
<td>Pattern value (C) of 2x2 pattern dither</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced only when pdth_sel is set to VDC5_PDTH_MD_2X2.</td>
</tr>
<tr>
<td>uint8_t pdth_pd</td>
<td>1</td>
<td>Pattern value (D) of 2x2 pattern dither</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Referenced only when pdth_sel is set to VDC5_PDTH_MD_2X2.</td>
</tr>
</tbody>
</table>
2.3.23 R_VDC5_GammaCorrection

Synopsis  
Gamma correction setup

Header  
r_vdc5.h

Declaration  
vdc5_error_t R_VDC5_GammaCorrection(
    const vdc5_channel_t   ch,
    const vdc5_onoff_t   gam_on,
    const vdc5_gamma_correction_t * const param);

Arguments  

• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1

• vdc5_onoff_t gam_on: Gamma correction ON/OFF setting

• vdc5_gamma_correction_t * param: Gamma correction setup parameter

Return value  

• vdc5_error_t: Error code
  — VDC5_OK: Normal termination
  — VDC5_ERR_PARAM_CHANNEL: Channel invalid error
  — VDC5_ERR_PARAM_BIT_WIDTH: Bit width error
  — VDC5_ERR_RESOURCE_OUTPUT: Output resource error

Details

(1) **Function**

This function performs the following processing for gamma correction:

- Turns on and off gamma correction processing.
- Sets up the gamma correction gain adjustment values for the G/B/R signals.
- Sets up the gamma correction start threshold values for the G/B/R signals.

The setup of the gamma correction parameters and the ON/OFF control of gamma correction processing can be made separately. The gamma correction parameter settings once established by this function remain valid until a hardware reset occurs or they are overwritten with other settings.

(2) **Use conditions**

Before using this function, it is necessary to set up display output by calling the function R_VDC5_DisplayOutput. The function returns an output resource error (VDC5_ERR_RESOURCE_OUTPUT) if the display output is not set up.

(3) **Parameter details**

The members of vdc5 Gamma_correction_t structure are described below.

```c
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;
} vdc5_gamma_correction_t;
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>const uint16_t *</td>
<td>gam_g_gain</td>
<td>Gain adjustment of area 0 to 31 of G signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 2047[LSB], 1024[LSB] = 1.0[time])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const uint8_t *</td>
<td>gam_g_th</td>
<td>Start threshold of area 1 to 31 of G signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 255[LSB])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const uint16_t *</td>
<td>gam_b_gain</td>
<td>Gain adjustment of area 0 to 31 of B signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 2047[LSB], 1024[LSB] = 1.0[time])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const uint8_t *</td>
<td>gam_b_th</td>
<td>Start threshold of area 1 to 31 of B signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 255[LSB])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const uint16_t *</td>
<td>gam_r_gain</td>
<td>Gain adjustment of area 0 to 31 of R signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 2047[LSB], 1024[LSB] = 1.0[time])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
<tr>
<td>const uint8_t *</td>
<td>gam_r_th</td>
<td>Start threshold of area 1 to 31 of R signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unsigned (0 to 255[LSB])</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify NULL when this parameter value need not be changed.</td>
</tr>
</tbody>
</table>

If the parameters have never been set up after a hardware reset, their initial values that are defined in the hardware manual remain valid. The initial values are given below.

- Gain adjustment of area 0 to 31 of G/B/R signal: All 1024 (= 1.0[time])
- Start threshold of area 1 to 31 of G/B/R signal: The start threshold value of area n is n x 8.
### 2.3.24 R_VDC5_GetISR

#### Synopsis
Interrupt service routine acquisition processing

#### Header
r_vdc5.h

#### Declaration
```c
void (*R_VDC5_GetISR(
    const vdc5_channel_t ch,
    const vdc5_int_type_t type)
    (const uint32_t int_sense);
```

#### Arguments
- `vdc5_channel_t ch`: Channel
  - `VDC5_CHANNEL_0`: Channel 0
  - `VDC5_CHANNEL_1`: Channel 1
- `vdc5_int_type_t type`: Interrupt type
  See 2.2.2(8) for details.

#### Return value
- `void (*)(const uint32_t int_sense)`: Function pointer to the interrupt service routine
  - Other than 0: Normal termination
  - 0: Error

#### Details

1. **Function**
   - This function returns the function pointer to the specified interrupt service routine.
   - It returns a '0' if the channel specified in `ch` or the interrupt type specified in `type` is found invalid.

2. **Use conditions**
   - There are no particular conditions with respect to the call of this function.
3. Sample Application

3.1 Specifications

Table 3-1 lists the peripheral functions to be used and their uses.

<table>
<thead>
<tr>
<th>Peripheral functions</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video display controller 5 (VDC5)</td>
<td>Controls the display of the LCD panel and sampling of video signals.</td>
</tr>
<tr>
<td>Channel 0</td>
<td></td>
</tr>
<tr>
<td>Digital video decoder (VDEC)</td>
<td>Decodes the composite signals of the video input.</td>
</tr>
<tr>
<td>Channel 0</td>
<td></td>
</tr>
<tr>
<td>Interrupt controller (INTC)</td>
<td>VDC5 channel 0 interrupt Specified line signal for panel output in graphics 3</td>
</tr>
<tr>
<td>On-chip large-capacity RAM</td>
<td>VRAM used by the VDC5</td>
</tr>
<tr>
<td>Serial communication interface with FIFO (SCIF)</td>
<td>For console output</td>
</tr>
<tr>
<td>Channel 2</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Operation Check Conditions

The sample code contained in this application note has been checked under the conditions listed below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcomputer used</td>
<td>RZ/A1H, CPU: ARM9</td>
</tr>
<tr>
<td>Operating frequency [MHz]</td>
<td>CPU clock: 400.0&lt;br&gt; Image processing clock: 266.67&lt;br&gt; Internal bus clock: 133.33&lt;br&gt; Peripheral clock 0: 33.33&lt;br&gt; Peripheral clock 1: 66.67</td>
</tr>
<tr>
<td>Operating voltage [V]</td>
<td>Power supply voltage (I/O): 3.3&lt;br&gt; Power supply voltage (internal): 1.18</td>
</tr>
<tr>
<td>Development environment</td>
<td>ARM Development Studio 5 (DS-5™) Version 5.16</td>
</tr>
<tr>
<td>Compiler</td>
<td>ARM C/C++ Compiler/Linker/Assembler , 5.03 [Build 102]</td>
</tr>
<tr>
<td>Compiler options (excluding the include path)</td>
<td>-O3 -Ospace --cpu=Cortex-A9 --littleend --arm --apcs=/interwork --no_unaligned_access --fpuv=vfpv3_fp16 -g --asm</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Boot mode 0 (CS0 area, 16-bits bus width)</td>
</tr>
<tr>
<td>Board to be used</td>
<td>R7S72100 CPU Board (Part number: RTK772100BC00000BR)</td>
</tr>
<tr>
<td></td>
<td>R7S72100 Optional Board (Part number: RTK772100B00000BR)</td>
</tr>
<tr>
<td>Device used</td>
<td>LCD monitor, Analog RGB output (Optional board: J15)&lt;br&gt; Video input interface (RCA connector, CPU board: J1)&lt;br&gt; Serial interface (D-Sub 9-pin connector, CPU board: J17)</td>
</tr>
</tbody>
</table>
Figure 3-1 shows the operating environment.

![Diagram of the operating environment](image)

**R7S72100 CPU Board & Optional Board**

3.3 **Related Application Note**

The application note that is related to this application note is listed below for reference.

- RZ/A1H Group Example of Initialization (R01AN1864EJ)
- RZ/A1H Group Digital Video Decoder Sample Driver (R01AN1823EJ)

3.4 **Description of Software**

3.4.1 **System Outline**

This sample program displays images on an LCD panel using the VDC5 driver. It supports the following display modes:

- **Graphics display:**
  Displays the images generated by the sample program on an LCD panel.
- **Video display:**
  Decodes the composite signal input with a digital video decoder (VDEC). The decoded video image is synthesized with the image that is generated by the sample program before being displayed on the LCD panel.

To switch between these two display modes, it is necessary to modify the program. The location to be modified is found in the function main in the source file named "main.c." See the source code shown below.
```c
int_t main(void) {
  #if defined(__ICCARM__) || defined(__GNUC__)
    SystemInit();
  #endif

  /* ==== Setup the I/O without buffering ==== */
  #ifdef __ICCARM__
    setvbuf( stdout, NULL, _IONBF, 0 );
    setvbuf( stdin, NULL, _IONBF, 0 );
  #endif

  printf("RZ/A1H CPU Board Sample Program. Ver.1.00\n");
  printf("Copyright (C) 2014 Renesas Electronics Corporation. All rights reserved.\n");

  printf("RZ/A1H Graphics Sample Program.\n Version %s\n", GRAPHICS_SAMPLE_VERSION);
  #if 1     /* Modify here to switch the display mode */
    printf("/_/_/_/_/ Graphics Sample _/_/_/_/\n");
    GRAPHICS_GraphicsSample();
  #else
    printf("/_/_/_/_/ Video Sample _/_/_/_/\n");
    GRAPHICS_VideoSample();
  #endif

  return 0;
}
```

Initially, the graphics display mode is turned on.

This sample program performs console I/O using a serial interface. It outputs on the console error information in addition to the version information which is generated within the above-mentioned function main. The baud rate of the serial interface is 115200 bps.

### 3.4.2 Memory Mappings

For the memory map and sections of this software, refer to the related application note entitled "RZ/A1H Group Example of Initialization". The section that is specific to this software is given below.

<table>
<thead>
<tr>
<th>Area Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VRAM</td>
<td>ZI Data</td>
<td>Used as the frame buffer for storing graphics and input images. It is allocated to the non-cache area in the on-chip large-capacity RAM.</td>
</tr>
</tbody>
</table>

### 3.4.3 Interrupts

Table 3-4 shows interrupts for the Sample Code.

<table>
<thead>
<tr>
<th>Interrupt (ID)</th>
<th>Priority</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDC5 channel 0 (GR3_VLINE0:78)</td>
<td>5</td>
<td>Specified line signal for panel output in graphics 3. Occurs at 60 Hz frequency which is the same as the refresh rate of the LCD panel.</td>
</tr>
</tbody>
</table>
3.5 Software Details

See the sample code for the definitions of the constants, structures, and variables that are used in the sample application.

3.5.1 List of Functions

A list of functions used in the sample code is summarized in Table 3-5. This document describes major functions only and does not cover all of the functions of the sample program.

<table>
<thead>
<tr>
<th>Function name</th>
<th>Section</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAPHICS_GraphicsSample</td>
<td>3.5.2(1)</td>
<td>Graphics display sample</td>
</tr>
<tr>
<td>GRAPHICS_VideoSample</td>
<td>3.5.2(2)</td>
<td>Video display sample</td>
</tr>
<tr>
<td>GRAPHICS_CreateFillColorImage</td>
<td>3.5.2(3)</td>
<td>Filled color image creation processing</td>
</tr>
<tr>
<td>GRPDRV_Init</td>
<td>3.5.2(4)</td>
<td>Graphics initialization processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_Term</td>
<td>3.5.2(5)</td>
<td>Graphics termination processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_GraphicsCreateSurface</td>
<td>3.5.2(6)</td>
<td>Graphics surface creation processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_VideoCreateSurface</td>
<td>3.5.2(7)</td>
<td>Video surface creation processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_DestroySurfaces</td>
<td>3.5.2(8)</td>
<td>Surface destruction processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_StartSurfaces</td>
<td>3.5.2(9)</td>
<td>Surface start processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRPDRV_StopSurfaces</td>
<td>3.5.2(10)</td>
<td>Surface stop processing (VDC5 driver wrapper function)</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdPanel</td>
<td>3.5.2(11)</td>
<td>LCD panel I/O port setup processing</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdTconSettings</td>
<td>3.5.2(12)</td>
<td>LCD TCON setup parameter acquisition processing</td>
</tr>
<tr>
<td>GRAPHICS_GetLvdsParam</td>
<td>3.5.2(13)</td>
<td>LVDS-related parameter acquisition processing</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdPanel_Ch0</td>
<td>3.5.2(14)</td>
<td>LCD panel I/O port setup processing (VDC5 channel 0)</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdPanel_Ch1</td>
<td>3.5.2(15)</td>
<td>LCD panel I/O port setup processing (VDC5 channel 1)</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdTconSettings_Ch0</td>
<td>3.5.2(16)</td>
<td>LCD TCON setup parameter acquisition processing (VDC5 channel 0)</td>
</tr>
<tr>
<td>GRAPHICS_SetLcdTconSettings_Ch1</td>
<td>3.5.2(17)</td>
<td>LCD TCON setup parameter acquisition processing (VDC5 channel 1)</td>
</tr>
</tbody>
</table>
3.5.2 Function Specifications

(1) GRAPHICS_GraphicsSample

Synopsis
Graphics display sample

Header
graphics.h

Declaration
void GRAPHICS_GraphicsSample(void);

Description
This is a sample function that performs various processing for graphics display. It displays the image that the application created using layer 2 (graphics 2) of VDC5 channel 0. The color format of the image is RGB565. See 3.5.3 for details.

Arguments
• None

Return value
• None

(2) GRAPHICS_VideoSample

Synopsis
Video display sample

Header
graphics.h

Declaration
void GRAPHICS_VideoSample(void);

Description
This is a sample function that performs various processing for video display. It displays the video input image from channel 0 using layer 0 (graphics 0) of VDC5 channel 0. It also superimposes the image that the application created using layer 2 (graphics 2) on the video input image for display. The buffer video-signal writing format is YCbCr422 and the color format for the created image is ARGB8888. See 3.5.3 for details.

Arguments
• None

Return value
• None

(3) GRAPHICS_CreateFillColorImage

Synopsis
Filled color image creation processing

Header
graphics.h

Declaration
void GRAPHICS_CreateFillColorImage(
  void
  * const buff,
  * const clut,
  const uint16_t width,
  const uint16_t height,
  const uint16_t stride,
  const vdc5_gr_format_t format,
Description

This function fills the specified buffer in a rectangular form with the specified size and format.
The rectangle is surrounded by a 1-pixel frame in the reverse color of the filled image.
There are 13 colors of fill color patterns which can be specified by their index. When `-1` is specified, the fill color is changed each time this function is called.

Arguments

- `void * buff`: Frame buffer address
- `uint32_t * clut`: Address of area containing CLUT
  Specify NULL if not required.
- `uint16_t width`: Width of the image to be created [pixels]
- `uint16_t height`: Height of the image to be created [lines]
- `uint16_t stride`: Frame buffer stride [bytes]
- `vdc5_gr_format_t gr_format`: Format of the frame buffer read signal
  - VDC5_GR_FORMAT_RGB565 (0): RGB565
  - VDC5_GR_FORMAT_RGB888 (1): RGB888
  - VDC5_GR_FORMAT_ARGB1555 (2): ARGB1555
  - VDC5_GR_FORMAT_ARGB4444 (3): ARGB4444
  - VDC5_GR_FORMAT_ARGB8888 (4): ARGB8888
  - VDC5_GR_FORMAT_CLUT8 (5): CLUT8
  - VDC5_GR_FORMAT_CLUT4 (6): CLUT4
  - VDC5_GR_FORMAT_CLUT1 (7): CLUT1
  - VDC5_GR_FORMAT_YCBCR422 (8): YCbCr422
  - VDC5_GR_FORMAT_YCBCR444 (9): YCbCr444
  - VDC5_GR_FORMAT_RGBA5551 (10): RGBA5551
  - VDC5_GR_FORMAT_RGBA8888 (11): RGBA8888
- `int32_t index`: Color index
  - 0 to 12: Color number
  - -1: Automatic change

Return value

- None
Arguments

- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_onoff_t vd_in_0: Video input channel 0
  - VDC5_OFF (0): Do not use.
  - VDC5_ON (1): Use.
- vdc5_onoff_t vd_in_1: Video input channel 1
  - VDC5_OFF (0): Do not use.
  - VDC5_ON (1): Use.

Return value

- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

(5) GRPDRV_Term

Synopsis

Graphics termination processing (VDC5 driver wrapper function)

Header

graphics_drv_wrapper.h

Declaration

vdc5_error_t GRPDRV_Term (const vdc5_channel_t ch);

Description

This is a wrapper function for the VDC5 driver. It is used to perform termination processing.
Inside this function, the following VDC5 driver's API function is called:
- R_VDC5_Terminate

Arguments

- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1

Return value

- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

(6) GRPDRV_GraphicsCreateSurface

Synopsis

Graphics surface creation processing (VDC5 driver wrapper function)

Header

graphics_drv_wrapper.h

Declaration

vdc5_error_t GRPDRV_GraphicsCreateSurface(
    const vdc5_channel_t ch,
    const vdc5_layer_id_t layer_id,
    void * const framebuff,
    const uint32_t fb_stride,
    const vdc5_gr_format_t gr_format,
    const vdc5_period_rect_t * const period_rect);

Description

This is a wrapper function for the VDC5 driver. It is used to make settings for controlling
data reads for graphics display.

Inside this function, the following VDC5 driver's API function is called:

- **R_VDC5_ReadDataControl**

**Arguments**

- **vdc5_channel_t ch**: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- **vdc5_layer_id_t layer_id**: Layer ID
  - VDC5_LAYER_ID_0_RD: Layer 0 read processing
  - VDC5_LAYER_ID_1_RD: Layer 1 read processing
  - VDC5_LAYER_ID_2_RD: Layer 2 read processing
  - VDC5_LAYER_ID_3_RD: Layer 3 read processing
- **void * framebuff**: Base address of the frame buffer
- **uint32_t fb_stride**: Line offset address of the frame buffer
- **vdc5_gr_format_t gr_format**: Format of the frame buffer read signal
  - VDC5_GR_FORMAT_RGB565 (0): RGB565
  - VDC5_GR_FORMAT_RGB888 (1): RGB888
  - VDC5_GR_FORMAT_ARGB1555 (2): ARGB1555
  - VDC5_GR_FORMAT_ARGB4444 (3): ARGB4444
  - VDC5_GR_FORMAT_ARGB8888 (4): ARGB8888
  - VDC5_GR_FORMAT_CLUT8 (5): CLUT8
  - VDC5_GR_FORMAT_CLUT4 (6): CLUT4
  - VDC5_GR_FORMAT_CLUT1 (7): CLUT1
  - VDC5_GR_FORMAT_YCBCR422 (8): YCbCr422 *
  - VDC5_GR_FORMAT_YCBCR444 (9): YCbCr444 *
  - VDC5_GR_FORMAT_RGBA5551 (10): RGBA5551
  - VDC5_GR_FORMAT_RGBA8888 (11): RGBA8888
- **vdc5_period_rect_t * period_rect**: Graphics display area

**Return value**

- **vdc5_error_t**: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

Note: YCbCr422 and YCbCr444 can be specified only for graphics 0 and 1.

(7) **GRPDRV_VideoCreateSurface**

**Synopsis**

Video surface creation processing (VDC5 driver wrapper function)

**Header**

`graphics_drv_wrapper.h`

**Declaration**

```c
vdc5_error_t GRPDRV_VideoCreateSurface(
    const vdc5_channel_t  ch,
    const vdc5_layer_id_t  layer_id,
    void          * const framebuff,
    const uint32_t   fb_stride,
    const vdc5_res_md_t  res_md,
    const vdc5_period_rect_t  * const res,
    const vdc5_period_rect_t  * const period_rect);
```

**Description**

This is a wrapper function for the VDC5 driver. It is used to make settings for controlling data writes and reads for video display.
Inside this function, the following VDC5 driver's API functions are called:

- R_VDC5_WriteDataControl
- R_VDC5_ReadDataControl

**Arguments**

- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- `vdc5_layer_id_t layer_id`: Layer ID
  - VDC5_LAYER_ID_0_WR: Layer 0 write processing
  - VDC5_LAYER_ID_1_WR: Layer 1 write processing
- `void * framebuff`: Base address of the frame buffer
- `uint32_t fb_stride`: Line offset address of the frame buffer
- `vdc5_res_md_t res_md`: Frame buffer video-signal writing format
  - VDC5_RES_MD_YCBCR422 (0): YCbCr422
  - VDC5_RES_MD_RGB565 (1): RGB565
  - VDC5_RES_MD_RGB888 (2): RGB888
  - VDC5_RES_MD_YCBCR444 (3): YCbCr444
- `vdc5_period_rect_t * res`: Image area to be captured
- `vdc5_period_rect_t * period_rect`: Graphics display area

**Return value**

- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

---

### GRPDRV_DestroySurfaces

**Synopsis**

Surface destruction processing (VDC5 driver wrapper function)

**Header**

`graphics_drv_wrapper.h`

**Declaration**

```
vdc5_error_t GRPDRV_DestroySurfaces(const vdc5_channel_t ch);
```

**Description**

This is a wrapper function for the VDC5 driver. It is used to clear the existing data write and read control settings.

Inside this function, the following VDC5 driver's API function is called:

- R_VDC5_ReleaseDataControl

**Arguments**

- `vdc5_channel_t ch`: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1

**Return value**

- `vdc5_error_t`: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

---

### GRPDRV_StartSurfaces

**Synopsis**

Surface start processing (VDC5 driver wrapper function)

**Header**

`graphics_drv_wrapper.h`
**vdc5_error_t GRPDRV_StartSurfaces(const vdc5_channel_t ch, const vdc5_gr_disp_sel_t * const gr_disp_sel);**

**Description**
This is a wrapper function for the VDC5 driver. It is used to start the processing that is set up by the existing data write and read control settings. Inside this function, the following VDC5 driver's API function is called:
- R_VDC5_StartProcess

**Arguments**
- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1
- vdc5_gr_disp_sel_t * gr_disp_sel: Graphics display mode

**Return value**
- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

---

**vdc5_error_t GRPDRV_StopSurfaces(const vdc5_channel_t ch);**

**Description**
This is a wrapper function for the VDC5 driver. It is used to stop the currently executing data write and read processing. Inside this function, the following VDC5 driver's API function is called:
- R_VDC5_StopProcess

**Arguments**
- vdc5_channel_t ch: Channel
  - VDC5_CHANNEL_0: Channel 0
  - VDC5_CHANNEL_1: Channel 1

**Return value**
- vdc5_error_t: Error code
  - VDC5_OK: Normal termination
  - Other than VDC5_OK: Error

---

**void GRAPHICS_SetLcdPanel(const vdc5_channel_t channel);**

**Description**
This function makes specific settings that are necessary for LCD panel output through the specified channel. This function calls the following functions that set up the LCD panel I/O
ports associated with the channels:
• GRAPHICS_SetLcdPanel_Ch0: Channel 0
• GRAPHICS_SetLcdPanel_Ch1: Channel 1

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1

Return value
• None

(12) GRAPHICS_SetLcdTconSettings

Synopsis
LCD TCON setup parameter acquisition processing

Header
lcd_panel.h

Declaration
void GRAPHICS_SetLcdTconSettings(
    const vdc5_channel_t channel,
    const vdc5_lcd_tcon_timing_t * * const outctrl);

Description
This function gets the LCD TCON timing setup data for the specified channel. It calls the following LCD TCON setup parameter acquisition functions for the associated channels:
• GRAPHICS_SetLcdTconSettings_Ch0: Channel 0
• GRAPHICS_SetLcdTconSettings_Ch1: Channel 1

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1
• vdc5_lcd_tcon_timing_t * * outctrl: Address of the area for storing the LCD TCON timing setup data table

Return value
• None

(13) GRAPHICS_GetLvdsParam

Synopsis
LVDS-related parameter acquisition processing

Header
lcd_panel.h

Declaration
vdc5_lvds_t * GRAPHICS_GetLvdsParam(const vdc5_channel_t channel);

Description
This function is used to get the LVDS-related parameters which become necessary when making use of the LVDS PLL as the panel clock.

Arguments
• vdc5_channel_t ch: Channel
  — VDC5_CHANNEL_0: Channel 0
  — VDC5_CHANNEL_1: Channel 1

Return value
• vdc5_lvds_t *: Address of the area containing the LVDS-related parameters
(14) **GRAPHICS_SetLcdPanel_Ch0**

**Synopsis**  
LCD panel I/O port setup (VDC5 channel 0)

**Header**  
lcd_analog_rgb_ch0.h  
lcd_lcd_kit_b01_ch0.h

**Declaration**  
void GRAPHICS_SetLcdPanel_Ch0(void);

**Description**  
This function sets up the I/O ports that are necessary for LCD panel output through channel 0.

**Arguments**  
• None

**Return value**  
• None

(15) **GRAPHICS_SetLcdPanel_Ch1**

**Synopsis**  
LCD panel I/O port setup (VDC5 channel 1)

**Header**  
lcd_analog_rgb_ch1.h  
lcd_lcd_kit_b01_ch1.h  
lcd_r0p7724le0011rl_ch1.h

**Declaration**  
void GRAPHICS_SetLcdPanel_Ch1(void);

**Description**  
This function sets up the I/O ports that are necessary for LCD panel output through channel 1.

**Arguments**  
• None

**Return value**  
• None

(16) **GRAPHICS_SetLcdTconSettings_Ch0**

**Synopsis**  
LCD TCON setup parameter acquisition processing (VDC5 channel 0)

**Header**  
lcd_analog_rgb_ch0.h  
lcd_lcd_kit_b01_ch0.h

**Declaration**  
void GRAPHICS_SetLcdTconSettings_Ch0(  
    const vdc5_lcd_tcon_timing_t * * const outctrl);

**Description**  
This function gets the LCD TCON timing setup data for channel 0.

**Arguments**  
• vdc5_lcd_tcon_timing_t ** outctrl:  
  Address of the area for storing the LCD TCON timing setup data table
(17) GRAPHICS_SetLcdTconSettings_Ch1

Synopsis
LCD TCON setup parameter acquisition processing (VDC5 channel 1)

Header
lcd_analog_rgb_ch1.h
lcd_lcd_kit_b01_ch1.h
lcd_r0p7724le0011rl_ch1.h

Declaration
void GRAPHICS_SetLcdTconSettings_Ch1(
    const vdc5_lcd_tcon_timing_t * * const outctrl);

Description
This function gets the LCD TCON timing setup data for channel 1.

Arguments
- vdc5_lcd_tcon_timing_t * * outctrl:
  Address of the area for storing the LCD TCON timing setup data table

Return value
- None
3.5.3 Flowcharts

Given below are the flowcharts of initialization, drawing, and termination processing for graphics and video display. Graphics display is carried out by the function GRAPHICS_GraphicsSample and video display by the function GRAPHICS_VideoSample. Both functions are called by the function main.

(1) Initialization processing for graphics display

Figure 3-2 shows the flowchart of initialization processing for graphics display.

![Figure 3-2 Initialization Processing for Graphics Display](image)

A description of the steps shown in Figure 3-2 follows.

1. Initializes the variables that are used inside the program.
2. Initializes the two frame buffers.
4. Performs initialization for graphics. 
   See 3.5.2(4) for details.
5. Gets the address of the back buffer (the buffer that is currently invisible) of the two frame buffers.
6. Writes image data into the back buffer.
   The image data is created by the function GRAPHICS_CreateFillColorImage. See 3.5.2(3) for details.
7. Sets up the VDC5 for graphics display.
   See 3.5.2(6) for details.
8. Calls the VDC5 driver's API function R_VDC5_CallbackISR.
   Registers the callback function associated with the specified line signal for panel output in graphics 3.
(2) **Initialization for video display**

Figure 3-3 shows the flowchart of initialization processing for video display.

![Flowchart of Initialization Processing for Video Display](image)

Note: G1 connects with G1 in Figure 3-4.

**Figure 3-3 Initialization Processing for Video Display**

A description of the steps shown in Figure 3-3 follows. Only the steps that differ from those for graphics display are explained here.

1. Initializes the video decoder.
   Refer to the related application note entitled "Digital Video Decoder Sample Driver (R01AN1823EJ)" for further information about this function.
2. Performs initialization for graphics.
   In video display mode, the channel of the video input to be used is specified with this function.
   See 3.5.2(4) for details.
3. Sets up the VDC5 for video display.
   See 3.5.2(7) for details.
4. Starts displaying
It is specified such that the video image from graphics 0 and the image from graphics 2 are superimposed for display. See 3.5.2(9) for details.

(3) Drawing processing

Figure 3-4 shows the flowchart of drawing processing for graphics display. The drawing processing for video display is identical to that for graphics display.

![Flowchart of Drawing Processing for Graphics Display](image)

**Figure 3-4 Drawing Processing for Graphics Display**

A description of the steps shown in Figure 3-4 follows.

1. Waits for vertical sync signals (Vsync).
   This program waits for 60 occurrences of Vsync. Since the refresh rate of the LCD panel is approximately 60Hz, the wait period for 60 Vsync signals is equivalent to approximately 1.0 second.

2. Switches between the two frame buffers.
   The front buffer and back buffer that are managed by the application are swapped.

3. Gets the address of the back buffer.

4. Writes image data into the back buffer.
   The image data is created by the function GRAPHICS_CreateFillColorImage. See 3.5.2(3) for details.

5. Calls the VDC5 driver's API function R_VDC5_ChangeReadProcess.
   To display the back buffer in which image data is written in step 4 above, designates the address of the back buffer as the frame buffer address.
   See 2.3.10 for details.

6. If no error is encountered during the draw processing, repeats the steps that have been taken so far (steps 1 to 5).

(4) Termination processing

Figure 3-5 shows the flowchart of termination processing for graphics display. The termination processing for video display is identical to that for graphics display.
Figure 3-5 Termination Processing for Graphics Display

A description of the steps shown in Figure 3-5 follows.

1. Executes the termination function.
The contents of this function are shown on the right side of the figure.
2. Calls the VDC5 driver's API function R_VDC5_CallbackISR.
   Removes the callback function associated with the specified line signal for panel output in graphics 3.
   See 2.3.6 for details.
3. Stops the display processing.
   See 3.5.2(10) for details.
4. Clears the VDC5 settings for display.
   See 3.5.2(8) for details.
5. Performs termination processing for graphics display.
   See 3.5.2(5) for details.

Note: G2 connects with G2 in Figure 3-4.
3.5.4 LCD Setup

(1) Setting up an LCD panel that uses the Hsync and Vsync signals

This sample program outputs images to the LCD monitor that is driven by the horizontal sync signal (Hsync) and vertical sync signal (Vsync). The specifications for the LCD panel are given in Table 3-6.

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock frequency</td>
<td>40.0 [MHz]</td>
</tr>
<tr>
<td>Horizontal period (Hp)</td>
<td>1056 [lines]</td>
</tr>
<tr>
<td>Horizontal sync signal width (Hsw)</td>
<td>128 [clocks]</td>
</tr>
<tr>
<td>Horizontal back porch (Hbp)</td>
<td>88 [clocks]</td>
</tr>
<tr>
<td>Horizontal display width (Hdp)</td>
<td>800 [clocks]</td>
</tr>
<tr>
<td>Horizontal front porch (Hfp)</td>
<td>40 [clocks]</td>
</tr>
<tr>
<td>Horizontal sync signal polarity</td>
<td>Positive</td>
</tr>
<tr>
<td>Vertical period (Vp)</td>
<td>628 [lines]</td>
</tr>
<tr>
<td>Vertical sync signal width (Vsw)</td>
<td>4 [lines]</td>
</tr>
<tr>
<td>Vertical back porch (Vbp)</td>
<td>23 [lines]</td>
</tr>
<tr>
<td>Vertical display width (Vdp)</td>
<td>600 [lines]</td>
</tr>
<tr>
<td>Vertical front porch (Vfp)</td>
<td>1 [line]</td>
</tr>
<tr>
<td>Vertical sync signal polarity</td>
<td>Positive</td>
</tr>
<tr>
<td>Data sampling</td>
<td>On rising edge of clock</td>
</tr>
<tr>
<td>Display format</td>
<td>RGB888</td>
</tr>
</tbody>
</table>

The settings listed in Table 3-7 must be made to generate signals that comply with the LCD panel specifications.

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Panel clock</td>
<td>LVDS PLL clock is used.</td>
</tr>
<tr>
<td></td>
<td>40.0 [MHz]</td>
</tr>
<tr>
<td>Horizontal sync signal pulse start position (Hss)</td>
<td>0 [clocks]</td>
</tr>
<tr>
<td>Horizontal sync signal pulse width</td>
<td>128 [clocks] (= Hsw)</td>
</tr>
<tr>
<td>Horizontal sync signal polarity</td>
<td>Non-inverted (positive polarity)</td>
</tr>
<tr>
<td>Vertical sync signal pulse start position (Vss)</td>
<td>624 x 2 [1/2 lines] (Vbp + Vdp + Vfp)</td>
</tr>
<tr>
<td>Vertical sync signal pulse width</td>
<td>4 x 2 [1/2 lines] (Vsw)</td>
</tr>
<tr>
<td>Vertical sync signal polarity</td>
<td>Non-inverted (positive polarity)</td>
</tr>
<tr>
<td>Data output</td>
<td>On falling edge of clock</td>
</tr>
<tr>
<td></td>
<td>The output is generated on the falling edge of the clock since the LCD panel data is sampled on the rising edge of the clock.</td>
</tr>
<tr>
<td>LCD panel output format</td>
<td>RGB888</td>
</tr>
</tbody>
</table>
See Figure 3-6 for the timing relationship between the relevant signals and effective data.

Figure 3-6  Hsync/Vsync Signal Driven LCD Panel Control Signal Timing

For the actual parameter settings, see also the following files in the sample program package:

- lcd_analog_rgb_ch0.c
- lcd_analog_rgb_ch0.h
- lcd_analog_rgb.h

Hsync and Vsync for the LCD panel start in Hss and Vss, respectively, after the VDC5’s internal reference sync signals. These are parameters for adjusting the timing; they may be set to '0' if not required. For this sample program, the horizontal signal start position Hss is set to '0'[clock cycles] and Vss to '624'[lines] (actual values are 624x2). The figure below illustrates the timing adjustment using Vss for this sample program.
A. \( V_{ss} = 0 \)

\( V_{sync} \) (VDC5)

\( V_{sync} \)

\( V_{ss} (0) \) \( \rightarrow \) \( V_{bp} (23) \) \( \rightarrow \) \( V_{dp} (600) \) \( \rightarrow \) \( V_{fp} (1) \)

\( V_{sw} (4) \) \( \rightarrow \) \( \text{Valid image data} \)

\( \text{res}_{f}.v_{ss} (27) \) \( \rightarrow \) \( \text{res}_{f}.v_{w} (600) \)

B. \( V_{ss} = 624 \)

\( V_{sync} \) (VDC5)

\( V_{sync} \)

\( V_{ss} (624) \) \( \rightarrow \) \( \text{Valid image data} \)

\( V_{bp} (23) \) \( \rightarrow \) \( \text{res}_{f}.v_{ss} (23) \) \( \rightarrow \) \( \text{res}_{f}.v_{w} (600) \) \( \rightarrow \) \( V_{fp} (5) \)

\( \text{Vsync} \) (VDC5): VDC5's internal reference vertical sync signal

\( \text{Hsync} \) (VDC5): VDC5's internal reference horizontal sync signal

\( \text{Vsync} \): vertical sync signal

\( \text{res}_{f}.v_{ss} \): Vertical enable signal start position for full screen

\( \text{res}_{f}.v_{w} \): Vertical enable signal width for full screen

\( v_{fp} \): Period from end of full screen vertical enable signal to vertical sync signal

Figure 3-7 shows the timings of the vertical sync signal without \( V_{ss} \) adjustment (\( V_{ss} = \)‘0’, A. in the figure) and with \( V_{ss} \) adjustment (\( V_{ss} = \)‘624’, B. in the figure).

When \( V_{ss} \) is set to ‘0’, the period (\( v_{fp} \)) from the end of the full screen vertical enable signal to the vertical sync signal is shorter than 4 lines, imposing a restriction on the effective period of the image (see Figure 2-6). This problem can be solved by setting \( V_{ss} \) to ‘624’, in which case the \( v_{fp} \) period turns to ‘5’.
(2) Setting up an LCD panel that uses the DE signal

An example of setting up an LCD panel that is driven by the data enable (DE) signal is described below. The specifications for this LCD panel are given in Table 3-8.

### Table 3-8 Specifications for the LCD Panel using the DE Signal

<table>
<thead>
<tr>
<th>Item</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock frequency</td>
<td>33.26 [MHz]</td>
</tr>
<tr>
<td>DE frame period (Vdp)</td>
<td>480 [lines]</td>
</tr>
<tr>
<td>DE frame blanking period</td>
<td>45 [lines]</td>
</tr>
<tr>
<td>Total DE frame period (Vp)</td>
<td>525 [lines] (= 480 + 45)</td>
</tr>
<tr>
<td>Total DE period (Hp)</td>
<td>1056 [clocks]</td>
</tr>
<tr>
<td>DE pulse period (Hdp)</td>
<td>800 [clocks]</td>
</tr>
<tr>
<td>DE signal polarity</td>
<td>Positive</td>
</tr>
<tr>
<td>Data sampling</td>
<td>Falling edge of clock</td>
</tr>
<tr>
<td>Display format</td>
<td>RGB666</td>
</tr>
</tbody>
</table>

In the VDC5, the DE signal is generated by the composition (logical product) of the vertical enable signal (VE) and horizontal enable signal (HE). The settings listed in Table 3-9 must be made to generate the DE signal that complies with the LCD panel specifications.

### Table 3-9 Settings for the LCD Panel Using the DE Signal

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Panel Clock</td>
<td>Peripheral clock 1 whose frequency is divided by 2</td>
</tr>
<tr>
<td></td>
<td>33.33 [MHz]</td>
</tr>
<tr>
<td>VE signal pulse start position (VEs)</td>
<td>10 x 2 [1/2 line]</td>
</tr>
<tr>
<td>VE signal pulse width (VEw)</td>
<td>480 x 2 [1/2 line]</td>
</tr>
<tr>
<td>HE signal pulse start position (HEs)</td>
<td>128 [clocks]</td>
</tr>
<tr>
<td>HE pulse width (HEw)</td>
<td>800 [clocks]</td>
</tr>
<tr>
<td>Data output</td>
<td>Rising edge of clock</td>
</tr>
<tr>
<td></td>
<td>The output is generated on the rising edge of the clock since the LCD panel data is sampled on the falling edge of the clock.</td>
</tr>
<tr>
<td>LCD panel output format</td>
<td>RGB666</td>
</tr>
</tbody>
</table>
See Figure 3-8 for the timing relationship between the relevant signals and effective data.

For the actual parameter settings, see also the following in the sample program package:

- lcd_r0p7724le0011rl_ch1.c
- lcd_r0p7724le0011rl_ch1.h
4. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

5. Related Documents

User's Manual
- R7S72100 CPU (GENMAI) Optional Board RTK772100B00000BR User's Manual
  The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
- The latest information can be downloaded from the Renesas Electronics website.

Application Note
- RZ/A1H Group Example of Initialization (R01AN1864EJ)
- RZ/A1H Group Digital Video Decoder Sample Driver (R01AN1823EJ)
  The latest version can be downloaded from the Renesas Electronics website.
Website and Support

Renesas Electronics Website
  http://www.renesas.com/

Inquiries
  http://www.renesas.com/contact/

All trademarks and registered trademarks are the property of their respective owners.
## Revision History

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Page</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>May. 23, 2014</td>
<td>-</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>
General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins
   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.
   - The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   - The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   - In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   - In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   - The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   - When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.
   - The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.