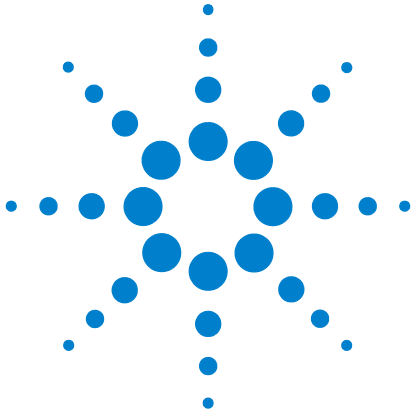


# Agilent ADCM-2700 VGA CMOS Camera Module

DRAFT 0.11

## Technical Reference Manual

Agilent Technologies

# Notices



## Printed

June 16, 2003

## Manual Part Number

00000-00000

## Edition

Preliminary edition, June 2003

Printed in USA

Agilent Technologies, Inc.
5301 Stevens Creek Blvd.
Santa Clara, CA 95051 USA

## Warranty

**The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.**

## Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

## Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

## Safety Notices

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

**A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.**

## In this Manual ...

This manual contains information on the use of the ADCM-2700 VGA CMOS Camera Module.

### 1 Theory of Operation

Provides information on the functions of the camera module, listing all registers for each function.

### 2 External Clocks or using the PLL

Information on what clock signals are needed by the camera module and how to use the optional PLL to change the operating frequency.

### 3 Serial Control Interface

Provides complete information on how to program and control the ADCM-2700 using the serial control interface.

### 4 Data Output Formats

Provides information of the various data output formats that can be programmatically selected.

### 5 Parallel Data Output Protocols

Provides information on the output protocols that are available using the parallel output port.

### 6 Synchronous Serial Data Output Protocols

Provides information on the output protocols that are available using the optional synchronous serial output port.

### 7 Simple Control Registers

Provides lists the registers that are most often used. Allows programming of the major camera functions.

### 8 Expert Hardware Registers

Provides expert hardware registers for fine control of camera operations. Data in these registers can be controlled by the simple control registers.

### 9 Expert Camera Controller Registers

Provides expert registers for fine control of camera operations. Data in these registers can be controlled by the simple control registers.

# Contents

## 7 Simple Control Registers



## 8 Expert Hardware Registers

## 9 Expert Camera Controller Registers

## 11 Expert Image Pipeline Registers

# 1 Theory of Operation

## Overview

The ADCM-2700 camera module contains a single integrated circuit which contains an image sensor and an image processor (sometimes called an image pipeline). The camera is controlled using a serial control interface to the IC. See Chapter 3, “Serial Control Interface,” starting on page 67 for information on how to send and receive control data to the image processor.

The ADCM-2700 uses a 2-wire serial port for programming and control, which is used to read and write data to the registers in both the image processor and the image sensor.

The following chapters contain detailed descriptions of all registers mentioned in this chapter:

- Chapter 7, “Simple Control Registers,” starting on page 137
- Chapter 8, “Expert Hardware Registers,” starting on page 165
- Chapter 9, “Expert Camera Controller Registers,” starting on page 179
- Chapter 10, “Expert Sensor Registers,” starting on page 269
- Chapter 11, “Expert Image Pipeline Registers,” starting on page 331

## Power On Sequence

Follow these steps to correctly power on the ADCM-2700:

1. Turn on $V_{CC}$ voltage (2.8 volts) and wait 20 milliseconds.
2. Turn on MCLK (13 MHz is the default value) and wait 150 milliseconds.
3. Read register 0x0004; if the return value equals 0x0001, power-up is complete.

**NOTE** Power must be applied before the clock. The 150 millisecond wait after the clock is applied is for the part to come out of reset and complete the initialization sequence.

Once power is turned on, the registers in the ADCM-2700 can be updated to optimize performance.

# Camera Module Block Diagram

The ADCM-2700 VGA CMOS camera module is a complete image processing system.

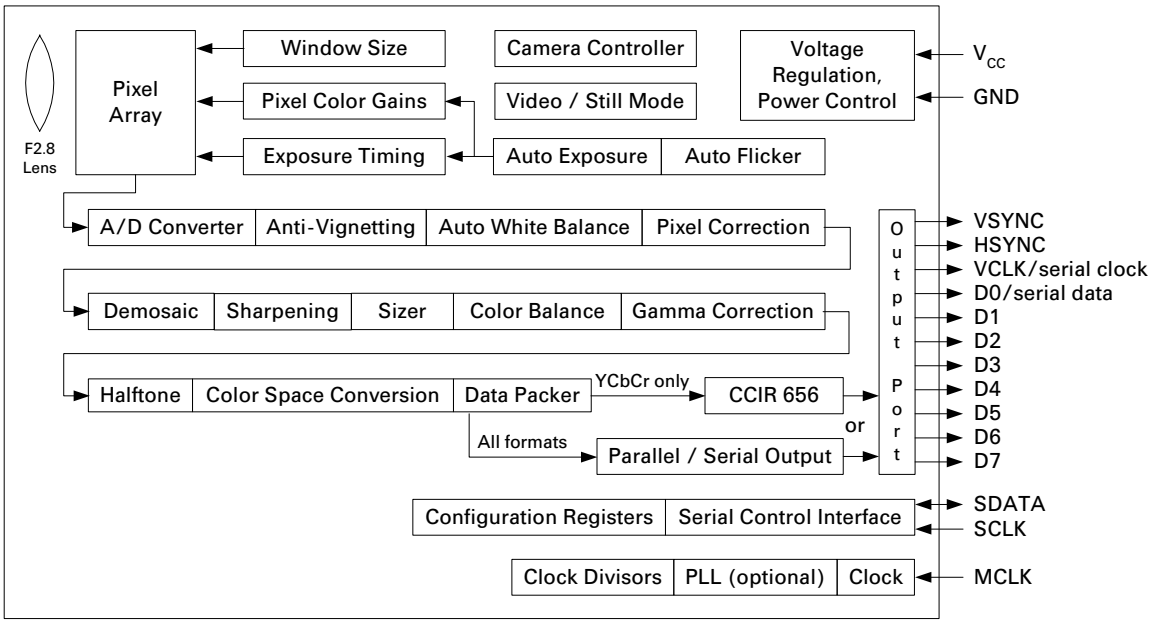



**Figure 1** ADCM-2700 block diagram

**Block diagram description**

| Feature | Description |
|---|---|
| A/D Converter | Converts the analog pixel voltages to 8-bit digital values |
| Anti-Vignetting | Variable gain to correct for light falloff in the corners of the picture |
| Auto Exposure | Adjusts the sensor to the amount of light present in the window using both exposure time and pixel gain |
| Auto Flicker | Keeps exposure times to multiples of the current flicker frequency, 50 or 60 Hz |
| Auto White Balance | Accommodates the slight color shifts that affect white in different kinds of light (daylight, fluorescent, incandescent). The camera module performs white balancing by digitally changing the gain-ratio of the red, green and blue channels and by adjusting the color-balancing matrix. White objects in the scene always look white in the final image. |
| Camera Controller | Overall functions of the camera module are centrally controlled |
| CCIR 656 Output | Parallel port using CCIR 656 formatted data. Data can be output with either external horizontal and vertical synchronization signals or using embedded synchronization codes. |

**Block diagram description (continued)**

| Feature | Description |
|---|---|
| Clock Divisors | Divisors to control the clock |
| Color Balance | Physical properties of optics dictate that images from the image sensor are not perfectly matched to the human eye; this block improves the color fidelity of the image and increases saturation |
| Color Space Conversion | Programmable color space conversion function to convert RGB values to a different color space. RGB values are multiplied by a 3 x 3 transform coefficient matrix and then offset. RGB to YCbCr is the default color space conversion. |
| Configuration Registers | Controls all camera module features |
| Data Packer | Data can be output in a variety of formats that use between 8 and 24 bits/pixel. |
| Demosaic | Image sensor produces a single red, green or blue pixel value for each location. The demosaic performs color interpolation to produce all three color components for each pixel location. The data is reduced to 8 bits per color per pixel at this stage. |
| Gamma Correction (Tonemap) | Pixel values acquired from the sensor are a linear function of the light present in the original scene. For computer monitors, the intensity produced by the display is a non-linear function of the pixel value. This non-linear relationship is characterized by a “gamma” curve. The gamma corrects the image data for display and can also make corrections to the contrast of the image. Conceptually, gamma correction is a 33-entry lookup table translating the linear response of the sensor into the non-linear characteristics of the display. |
| Halftone | Introduces pseudo random noise to bit reduced outputs to eliminate banding |
| Lens | High quality F/2.8 lens singlet |
| Output Port | Either synchronous serial or parallel output. Parallel output has eight data lines, two handshake lines and a video clock line. The synchronous serial has transmitted data and data clock lines. |
| Pixel Array | Sensor consists of a 640 x 480 pixel array, which can be windowed to any size between 640 x 480 and 24 x 24. The array can be mirrored in both the horizontal and vertical directions. The pixel array is actually 648 x 488, the extra pixels are used for the Bayer pattern needed at the edges of the picture for the demosaic. |
| Pixel Color Gain | Analog gain controlled by the auto exposure block |
| Pixel Correction | Circuit reduces the effects of pixel mismatch |
| PLL | Optional phase lock loop for fine clock control |
| Sharpening | Applies a variable sharpening filter to the image |

**Block diagram description (continued)**

| Feature | Description |
|---|---|
| Sizer | Allows the output image to be scaled from the sensor input. The scaling occurs after the demosaic operation and before the color balance. |
| Timing Control | Exposure control for the image sensor; exposure is in row times |
| Video / Still Mode | Video mode is nominally QVGA; still mode is VGA |
| Voltage Regulation | Internal voltage regulator |
| Window Size | Allows the sensor output to be windowed to any location on the sensor. Beginning and ending rows and columns can be specified, allowing the window to be any size, in any location. |

# Image Data Flow

The following table shows the flow of data from the sensor, through the image pipeline and out of the module.

**Image data flow**

| Operation | Description | Settings / Options |
|---|---|---|
| Image data from the sensor | Raw data from the sensor is input to the image pipeline | |
| Auto Exposure | Adjust sensor gain and exposure time to meet target average pixel luminance. | Enable/disable using the AF_CTRL1 register. See discussion of auto exposure and auto white balance functions for descriptions of configuration registers |
| Auto White Balance | Equalize average pixel luminance among color bands | |
| Auto Flicker | Adjust exposure time to eliminate flicker | |
| Anti-Vignetting | Corrects for light falloff in the corners of the picture | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers |
| Statistics | Collect image statistics such as peak values, pixel sums and histograms on a one to many frame basis. | Initialize and configure the statistics functions using the STAT_CAP_CTRL and STAT_MODE_CTRL registers. See "Image Statistics and Histograms" on page 45 for description of these outputs. NOTE: Usually controlled by firmware |
| Pixel Correction | Correct pixel values for mismatched pixels | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers |

**Image data flow (continued)**

| Operation | Description | Settings / Options |
|---|---|---|
| Demosaic | Convert raw Bayer pattern pixel data into red, green and blue image planes. | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers |
| Sharpening | Edges in the image are intensified | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers |
| Sizer | Resize the image | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers. Select the input and output height and width using the SRZ_IN and SZR_OUT registers |
| Color Balance | Adjust for the color filter response of the image sensor. | Use default or custom color correction matrices. |
| Gamma Correction | Apply a non-linear transfer function to the image data. | Enable/disable and choose linear or bottom-weighting using the PROC_CTRL_V or PROC_CTRL_S registers. Use default or custom table |
| Halftone | Add pseudo random noise to reduced bit depth images to eliminate banding. | Enable/disable using the PROC_CTRL_V or PROC_CTRL_S registers |
| Color Space Conversion | Convert RGB data to the desired color space | Use default (RGB to YCbCr) or custom conversion matrices |
| Data Packer | Determines data output format | 16 possible formats controlled by the OUTPUT_CTRL_V and OUTPUT_CTRL_S register. See “Data Output Formats" on page 105 for details |
| Data Output | Output using parallel, synchronous serial or CCIR | Many options. See “Parallel Data Output Protocols" on page 83 and “Synchronous Serial Output Protocols" on page 119 for details |

## Addressing RAM and Registers

The ADCM-2700 RAM and register space is divided into blocks of 64 words, each individually addressable. The module uses a 14-bit addressing scheme to reference blocks and positions within a block. Bits 13:7 of an address specify the base address of a block and bits 6:0 specify the address offset within a block. From each block address, the address offsets start at 0x00 and increment by 0x02 to 0x7e.

To properly address a RAM or register, read and write commands must specify the base address and the address offset of the targeted memory location. However, because the ADCM-2700 module stores the current base address, a read / write function does not need to rewrite the base address when reading/writing multiple locations within the same address block. After writing the starting base address, read / write functions must only specify the base address if it is different from the last read / write operation. See Chapter 3, “16-Bit Read and Write Examples,” starting on page 74 for example commands. Each data byte is sent most significant bit first, the data words are sent least significant byte first.

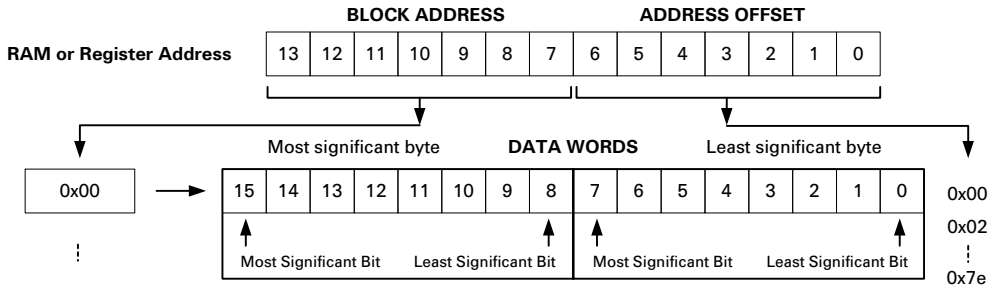



**Figure 2** Address block and offsets, bit order

**Addresses, full 16-bit and block number**

| Block Addresses | | | |
|---|---|---|---|
| **Decimal** | **Hex** | **Description** | **16-bit address** |
| 0 | 0x00 | Simple control registers | 0x0000 – 0x0026 |
| | | Reserved | 0x0028 – 0x007e |
| 1 | 0x01 | Expert hardware registers | 0x0080 – 0x0094 |
| | | Reserved | 0x0096 – 0x00fe |
| 2 | 0x02 | Expert camera control registers | 0x0100 – 0x017e |
| 3 | 0x03 | | 0x0180 – 0x01fe |
| 4 | 0x04 | | 0x0200 – 0x027e |
| 5 | 0x05 | | 0x0280 – 0x0292 |
| | | Reserved | 0x0294 – 0x02fe |

**Addresses, full 16-bit and block number (continued)**

| Block Addresses | | | |
|---|---|---|---|
| **Decimal** | **Hex** | **Description** | **16-bit address** |
| 6 – 15 | 0x06 – 0x0f | Reserved | 0x0300 – 0x07fe |
| 16 | 0x10 | Sensor registers (byte addressable) | 0x0800 – 0x0837 |
| | | Reserved (byte addressable) | 0x0838 – 0x087f |
| 17 | 0x11 | Reserved (byte addressable) | 0x0880 – 0x08d6 |
| | | Pixel timing (byte addressable) | 0x08d7 – 0x08f3 |
| | | Reserved | 0x08f4 – 0x08ff |
| 18 – 31 | 0x12 – 0x1f | Reserved | 0x0900 – 0x0ffe |
| 32 | 0x20 | Reserved | 0x1000 |
| | | Expert image pipeline registers | 0x1002 – 0x107e |
| 33 | 0x21 | | 0x1080 – 0x109c |
| | | Reserved | 0x109e – 0x10fe |
| 34 – 39 | 0x22 – 0x27 | Reserved | 0x1100 – 0x13fe |
| 40 | 0x28 | Tonemap RAM “All” | 0x1400 – 0x1440 |
| | | Reserved | 0x1442 – 0x147e |
| 41 | 0x29 | Tonemap RAM red | 0x1480 – 0x14c0 |
| | | Reserved | 0x14c2 – 0x147e |
| 42 | 0x2a | Tonemap RAM green | 0x1500 – 0x1540 |
| | | Reserved | 0x1542 – 0x157e |
| 43 | 0x2b | Tonemap RAM blue | 0x1580 – 0x15c0 |
| | | Reserved | 0x14c2 – 0x147e |
| 44 | 0x2c | Image statistics | 0x1600 – 0x167e |
| 45 | 0x2d | | 0x1680 – 0x16be |
| | | Reserved | 0x16c0 – 0x16fe |
| 46, 47 | 0x2e, 0x2f | Reserved | 0x1700 – 0x17fe |
| 48 | 0x30 | Anti-vignetting, red | 0x1800 – 0x183e |
| | | Anti-vignetting, green | 0x1840 – 0x187e |
| 49 | 0x31 | Anti-vignetting, blue | 0x1880 – 0x18be |

## Simple Control Registers

Since determining the register settings of the ADCM-2700 for a particular configuration such as a fixed frame rate can be complex, a set of simple control registers are present to simplify the process. For details of each register, see Chapter 7, “Simple Control Registers,” starting on page 137.

**Simple control registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| ID | 0x0000 | Chip ID | 0x0060 | 139 |
| CONTROL | 0x0002 | Camera control | 0x0001 | 140 |
| STATUS | 0x0004 | Camera status | 0x0004 | 142 |
| CLK_PER | 0x0006 | External clock frequency | 0x32c8 | 144 |
| SIZE | 0x0008 | Image size and orientation | 0x0605 | 145 |
| OUTPUT_FORMAT | 0x000a | Output format | 0x0909 | 147 |
| EXPOSURE | 0x000c | Exposure | 0x03e8 | 149 |
| EXP_ADJ | 0x000e | Exposure adjustment | 0x0000 | 150 |
| ILLUM | 0x0010 | Illumination | 0x0000 | 151 |
| FRAME_RATE | 0x0012 | Requested frame rate | 0x0096 | 152 |
| A_FRAME_RATE | 0x0016 | Actual frame rate | 0x0096 | 154 |
| SENSOR_WID_V | 0x0018 | User-defined sensor output width, video | 0x0000 | 155 |
| SENSOR_HGT_V | 0x001a | User-defined sensor output height, video | 0x0000 | 156 |
| OUTPUT_WID_V | 0x001c | User-defined picture output width, video | 0x0000 | 157 |
| OUTPUT_HGT_V | 0x001e | User-defined picture output height, video | 0x0000 | 158 |
| SENSOR_WID_S | 0x0020 | User-defined sensor output width, still | 0x0000 | 159 |
| SENSOR_HGT_S | 0x0022 | User-defined sensor output height, still | 0x0000 | 160 |
| OUTPUT_WID_S | 0x0024 | User-defined picture output width, still | 0x0000 | 161 |
| OUTPUT_HGT_S | 0x0026 | User-defined picture output height, still | 0x0000 | 162 |

### Using the Simple Control Registers

Follow these steps to change any of the simple control registers:

1. Stop the camera – write 0x0000 to the CONTROL register.
2. Change the relevant register.
   For example, change the output size from QCIF to CIF, write 0x0202 to the SIZE register.
3. Set the CONFIG bit in the CONTROL register, write 0x0004.
4. Wait for the CONFIG bit in CONTROL to clear.
5. Start the camera – write 0x0001 to the CONTROL register.

## Expert Hardware Registers

The following registers are programmed to calculated values by setting the CONFIG bit in the CONTROL register. After configuration, these registers can be changed to alter the camera operation. Under normal operation, these registers should not be changed.

**Simple control registers – expert**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CLK_DIV | 0x0080 | Clock divider for prescale of MCLK | 0x0001 | 166 |
| CTL_CLK_DIV | 0x0082 | Clock dividers for control and serial interface | 0x4000 | 167 |
| SEN_CLK_DIV | 0x0084 | Sensor clock dividers | 0x0000 | 168 |
| IP_CLK_DIV | 0x0086 | Image pipeline dividers | 0x0000 | 169 |
| TST_MODE | 0x0088 | Latched test mode | 0x0000 | 170 |
| SER_ADDR | 0x008a | Serial control interface device address | 0x0053 | 171 |
| SER_PARAM | 0x008c | Serial control interface parameters | 0x0000 | 172 |
| OUT_CTRL | 0x008e | Output control | 0x0000 | 173 |

## Image Integration using the Image Sensor

To begin an exposure, the top pixel row of the viewing window is reset. After reset, the row begins integration. After one row process time elapses, the next row is reset. This continues until the bottom row of the viewing window is reached.

In continuous run mode, this process repeats by wrapping to the top row of the viewing window. In single frame capture mode, the process ends when the bottom row of the viewing window is reached. In continuous run mode, if the integration time is less than the time to cycle through a frame there is no overhead time between frames. If the integration time is greater than the time to cycle through a frame, there is an overhead delay between frames equal to integration time minus the time to cycle through a frame.

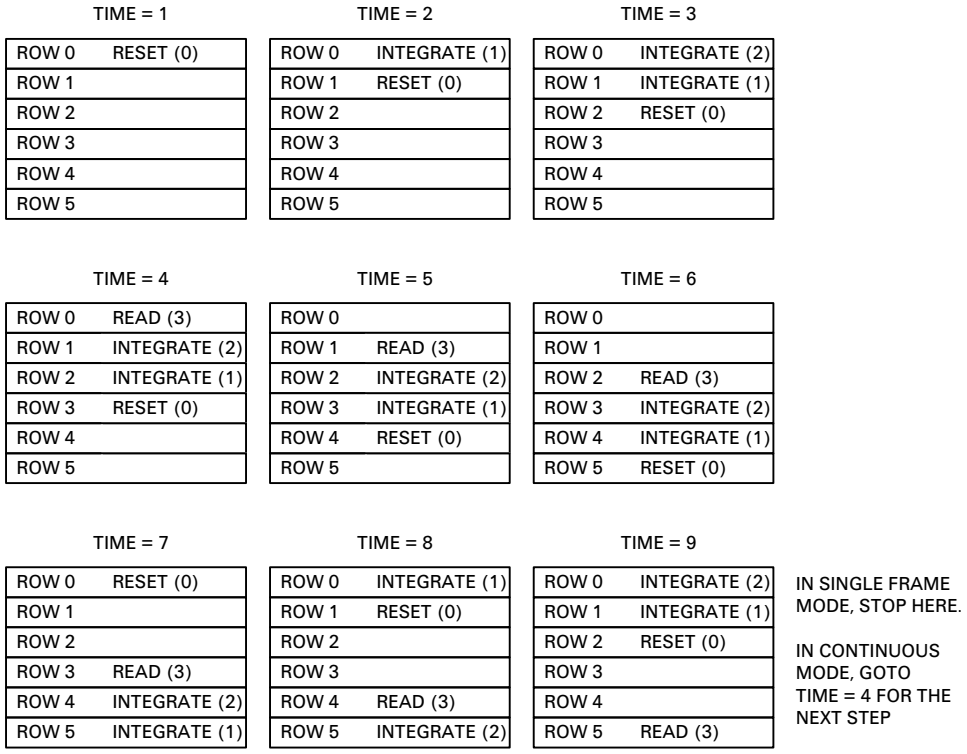



**Figure 3** Six row window with integration time = 2 rows

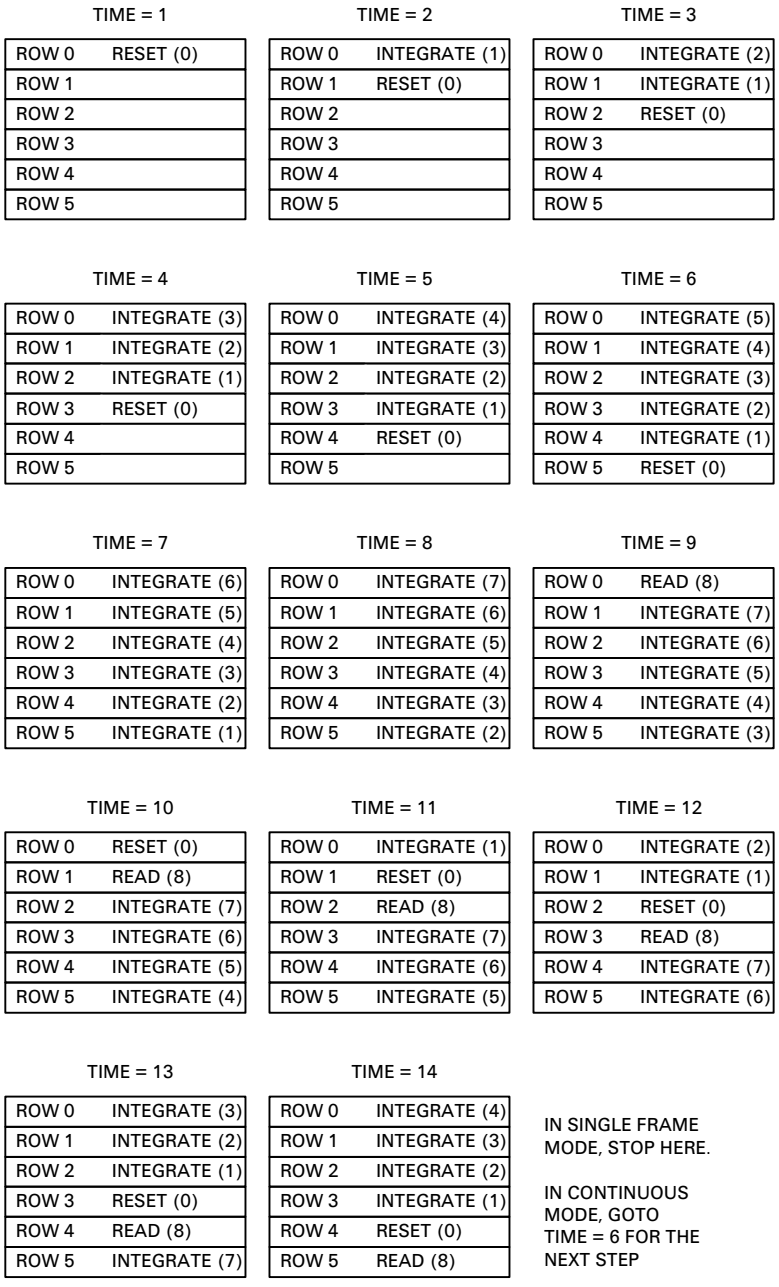



**Figure 4** Six row window with integration time = 7 rows

### Image Sensor Array

The image sensor has a rectangular pixel array of 640 x 480 pixels. The array can be windowed, allowing multiple output formats from one image sensor. The following registers in the sensor determine the active rows and columns:

**Sensor window**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| FWROW | 0x080a | First window row | 0x01 | 278 |
| FWCOL | 0x080b | First window column | 0x01 | 279 |
| LWROW | 0x080c | Last window row | 0x7a | 280 |
| LWCOL | 0x080d | Last window column | 0xa2 | 281 |

## Windowing and Panning

Data from the sensor is usually in well defined formats:

**Available window sizes**

| Mode | VGA | CIF | QVGA | QCIF | QQVGA | QQCIF | QQQVGA |
|---|---|---|---|---|---|---|---|
| Landscape | 640 x 480 | 352 x 288 | 320 x 240 | 176 x 144 | 160 x 120 | 88 x 72 | 80 x 60 |
| Portrait | Not available | 288 x 352 | 240 x 320 | 144 x 176 | 120 x 160 | 72 x 88 | 60 x 80 |

**NOTE** Since the lens covers the entire landscape VGA area, as the window size changes the field of view and the effective focal length of the lens changes.

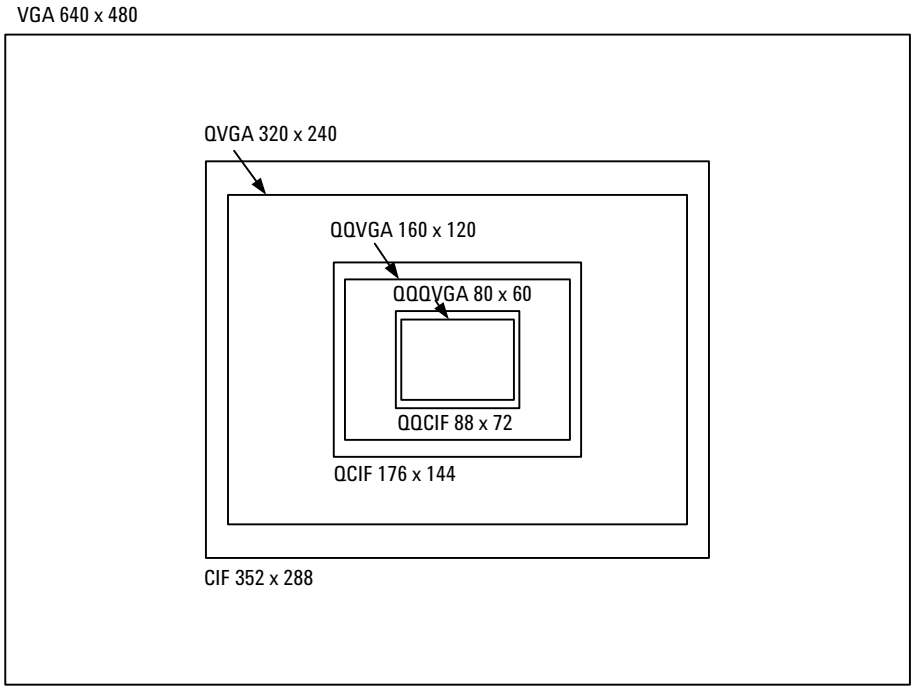



**Figure 5** Relative window sizes, landscape mode

Since the starting and stopping rows and columns can be individually controlled, windows smaller that 640 x 480 can be placed anywhere on the sensor.

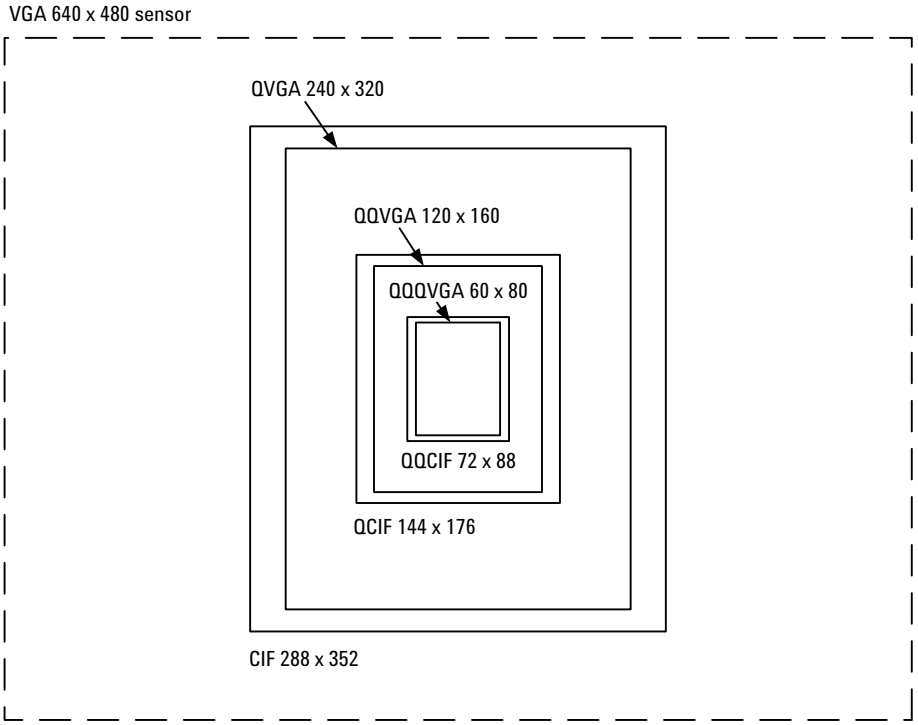



**Figure 6** Relative window sizes, portrait mode

## Window Sizes – Using the SIZE Register

When using the SIZE register, there are eight different output sizes available:

- VGA
- CIF
- QCIF
- QQCIF
- QVGA
- QQVGA
- QQQVGA
- User defined – See "User-Defined Window Sizes using the Simple Control Registers" on page 19.

## Manual Window Size using the Sizer

The output window can be controlled by manually setting the sensor window using the FWROW, FWCOL, LWROW and LWCOL registers and then programming the sizer for the wanted output size. The following examples use the largest area on the sensor possible (for the widest field of view).

### VGA, QVGA, QQVGA and QQQVGA

For VGA, QVGA, QQVGA and QQQVGA, the window size from the sensor is VGA, which has an aspect ratio of 1:1.3$\overline{3}$. The QVGA, QQVGA and QQQVGA outputs are achieved with the sizer (see “Sizer" on page 21).

**VGA window size**

| FWROW | FWCOL | LWROW | LWCOL |
|---|---|---|---|
| 0x01 | 0x01 | 0x7a | 0xa2 |

**VGA, QVGA, QQVGA and QQQVGA sizer values**

| Window Size | SZR_IN_W | SZR_IN_H | SZR_OUT_W | SZR_OUT_H |
|---|---|---|---|---|
| VGA | 0x0280 | 0x01e0 | 0x0280 | 0x01e0 |
| QVGA | 0x0280 | 0x01e0 | 0x0140 | 0x00f0 |
| QQVGA | 0x0280 | 0x01e0 | 0x00a0 | 0x0078 |
| QQQVGA | 0x0280 | 0x01e0 | 0x0050 | 0x003c |

### CIF, QCIF and QQCIF

For CIF, QCIF and QQCIF, the window size from the sensor has the same height as CIF, but the width is reduced to get the CIF aspect ratio of 1:1.2$\overline{2}$. All of the output final sizes are achieved with the sizer (see “Sizer" on page 21).

**CIF window size**

| FWROW | FWCOL | LWROW | LWCOL |
|---|---|---|---|
| 0x01 | 0x08 | 0x7a | 0x9b |

**CIF, QCIF and QQCIF sizer values**

| Window Size | SZR_IN_W | SZR_IN_H | SZR_OUT_W | SZR_OUT_H |
|---|---|---|---|---|
| CIF | 0x024a | 0x01e0 | 0x0160 | 0x0120 |
| QCIF | 0x024a | 0x01e0 | 0x00b0 | 0x0090 |
| QQCIF | 0x024a | 0x01e0 | 0x0058 | 0x0048 |

## Manual Window Size Not Using the Sizer

By selecting the active rows and columns, select different modes (VGA, CIF, QVGA, QCIF, QQVGA, QQCIF, QQQVGA) or any other value smaller than 640 x 480 and greater or equal to 24 x 24. If the selected format is smaller than the image array, it can be electronically panned by changing the active rows and columns.

Manual mode allows portrait mode for some of the smaller window sizes, something that is not available using the SIZE register. In manual mode, the sizer must be turned off (PROC_CTRL_V or PROC_CTRL_S register) or the sizer will overwrite the new window settings.

The following table lists the mode (centered in the image array) for each window:

**Window modes**

| Mode | Orientation | Size | FWROW | FWCOL | LWROW | LWCOL |
|---|---|---|---|---|---|---|
| VGA | Landscape | 640 x 480 | 0x01 | 0x01 | 0x7a | 0xa2 |
| CIF | Landscape | 352 x 288 | 0x19 | 0x25 | 0x62 | 0x7e |
| | Portrait | 288 x 352 | 0x11 | 0x2d | 0x6a | 0x76 |
| QVGA | Landscape | 320 x 240 | 0x1f | 0x29 | 0x5c | 0x7a |
| | Portrait | 240 x 320 | 0x15 | 0x33 | 0x66 | 0x70 |
| QCIF | Landscape | 176 x 144 | 0x2b | 0x3b | 0x50 | 0x68 |
| | Portrait | 144 x 176 | 0x27 | 0x3f | 0x54 | 0x64 |
| QQVGA | Landscape | 160 x 120 | 0x2e | 0x3d | 0x4d | 0x66 |
| | Portrait | 120 x 160 | 0x29 | 0x42 | 0x52 | 0x61 |
| QQCIF | Landscape | 88 x 72 | 0x34 | 0x46 | 0x47 | 0x5d |
| | Portrait | 72 x 88 | 0x32 | 0x48 | 0x49 | 0x5b |
| QQQVGA | Landscape | 80 x 60 | 0x35 | 0x47 | 0x45 | 0x5c |
| | Portrait | 60 x 80 | 0x33 | 0x49 | 0x48 | 0x59 |

**NOTE** The sensor requires that the row and column addresses begin on 4-byte boundaries; the register set forces this. Data written to the registers are actually bits [9:2]. For the first row and column registers, the lower two bits are forced to “00”. For the last row and column registers, the lower two bits are forced to “11”. This is taken care of in the formulas on the following page. The divide by 4 corrects for the two-bit shift and the 4 and 8 constants add in the extra Bayer pixels needed for demosaicing.

Use the following formulas to determine other centered image sizes:

### FWROW:

Determine the offset to center the window in the array. Subtract the number of rows from 480 and divide the result by 2. Add 4 to this number, divide by 4 and convert to hex.

$$FWROW = HEX\left(\frac{\frac{480 - rows}{2} + 4}{4}\right)$$

### FWCOL:

Determine the offset to center the window in the array. Subtract the number of columns from 640 and divide the result by 2. Add 4 to this number, divide by 4 and convert to hex.

$$FWCOL = HEX\left(\frac{\frac{640 - columns}{2} + 4}{4}\right)$$

### LWROW:

Determine the offset to center the window in the array. Subtract the number of rows from 480 and divide the result by 2. Add the total number of rows plus 8 to this number, divide by 4 and convert to hex.

$$LWROW = HEX\left(\frac{\frac{480 - rows}{2} + rows + 8}{4}\right)$$

### LWCOL:

Determine the offset to center the window in the array. Subtract the number of columns from 640 and divide the result by 2. Add the total number of columns plus 8 to this number, divide by 4 and convert to hex.

$$LWCOL = HEX\left(\frac{\frac{640 - columns}{2} + columns + 8}{4}\right)$$

## User-Defined Window Sizes using the Simple Control Registers

When the VSIZE or SSIZE bits in the SIZE register are 111, then the output size is defined by eight simple control registers.

**Simple control registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SENSOR_WID_V | 0x0018 | User-defined sensor output width, video | 0x0000 | 155 |
| SENSOR_HGT_V | 0x001a | User-defined sensor output height, video | 0x0000 | 156 |
| OUTPUT_WID_V | 0x001c | User-defined picture output width, video | 0x0000 | 157 |
| OUTPUT_HGT_V | 0x001e | User-defined picture output height, video | 0x0000 | 158 |
| SENSOR_WID_S | 0x0020 | User-defined sensor output width, still | 0x0000 | 159 |
| SENSOR_HGT_S | 0x0022 | User-defined sensor output height, still | 0x0000 | 160 |
| OUTPUT_WID_S | 0x0024 | User-defined picture output width, still | 0x0000 | 161 |
| OUTPUT_HGT_S | 0x0026 | User-defined picture output height, still | 0x0000 | 162 |

These registers will calculate the proper sensor register values, FWROW FWCOL, LWROW and LWCOL, as well as the sizer values for the wanted output size.

### Panorama Example

If the wanted final output size is 100 pixels by 320 pixels (a landscape panorama) and the full field angle of the lens is wanted, then the full width of the sensor needs to be used. If the image is not to be anamorphically stretched, the sizer needs to be the same for both the X and Y directions. Since the sensor is 640 pixels wide, the sizer factor is 320/640 or 0.5. The sensor height must be (output size)/sizer factor = 100/0.5 = 200. Assuming this is for video mode, write the following values to the registers:

**Panorama example**

| Register name | Decimal | Hexidecimal |
|---|---|---|
| SENSOR_HGT_V | 200 | 0x00c8 |
| SENSOR_WID_V | 640 | 0x0280 |
| OUTPUT_HGT_V | 100 | 0x0064 |
| OUTPUT_WID_V | 320 | 0x0140 |

Make sure that the VSIZE field of SIZE = 111.

# Subsampling

The ADCM-2700 camera module has the ability to subsample the image array to reduce the size of the image without changing the field of view. For better image quality, however, the use of the sizer is recommended.

Rows and columns can be subsampled with 2:1 decimation. With 2:1 row and column subsampling there is a 4:1 reduction in data. Subsampling is controlled using the VSUB and SSUB bits in the SIZE register. Once the camera module is configured, the needed data will be written to the sensor.

Alternatively, the VSUB bit in the SEN_CTRL_V register and the SSUB bit in the SEN_CTRL_S register can control subsampling. To use this mode, first disable the camera by writing 0x0000 to the CONTROL register (address 0x0002), change the SEN_CTRL register and then re-enable the camera by writing 0x0001 to CONTROL for video mode, or 0x0002 for still mode. The image processor will then write the needed data to the sensor registers.

**Subsampling registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SIZE | 0x0008 | Image size and orientation | 0x0605 | 145 |
| SEN_CTRL_V | 0x011c | Sensor control – video mode | 0x0000 | 200 |
| SEN_CTRL_S | 0x013c | Sensor control – still mode | 0x0000 | 218 |
| CONFIG_1 | 0x081b | Sensor configuration | 0x0e | 294 |
| CONFIG_2 | 0x0827 | Sensor configuration 2 | 0x00 | 298 |

**NOTE** DO NOT write data directly to the sensor because it will be overwritten by the camera controller.

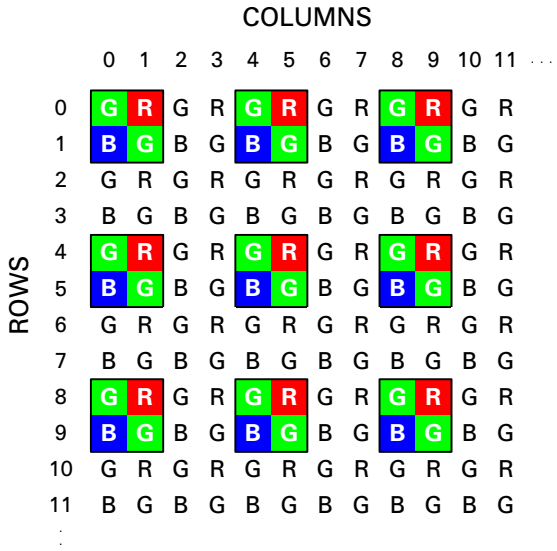



**Figure 7** Sensor subsampling

## Sizer

The ADCM-2700 camera module includes a “sizer” to resize the output. This allows the output format to change but the field of view to remain constant. Data from the sensor is usually in well defined formats (see “Available window sizes" on page 13).

The only requirements of the sizer is that the input image size must match the output size of the sensor and that the output image is smaller than the input image. Since the width and height are controlled with different registers, anamorphic size changes are possible.

The simple control register SIZE will automatically set the video and still mode sizer registers.

**Sizer registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SIZE | 0x0008 | Image size and orientation | 0x0605 | 145 |
| SZR_IN_WID_V | 0x0100 | Sizer input width – video mode | 0x0280 | 184 |
| SZR_IN_HGT_V | 0x0102 | Sizer input height – video mode | 0x01e0 | 185 |
| SZR_OUT_WID_V | 0x0104 | Sizer output width – video mode | 0x0140 | 186 |
| SZR_OUT_HGT_V | 0x0106 | Sizer output height – video mode | 0x00f0 | 187 |
| SZR_IN_WID_S | 0x0120 | Sizer input width – still mode | 0x0280 | 202 |
| SZR_IN_HGT_S | 0x0122 | Sizer input height – still mode | 0x01e0 | 203 |
| SZR_OUT_WID_S | 0x0124 | Sizer output width – still mode | 0x0280 | 204 |
| SZR_OUT_HGT_S | 0x0126 | Sizer output height – still mode | 0x01e0 | 205 |
| SZR_IN_W | 0x1020 | Sizer input width | 0x0280 | 352 |
| SZR_IN_H | 0x1022 | Sizer input height | 0x01e0 | 353 |
| SZR_OUT_W | 0x1024 | Sizer output width | 0x0140 | 354 |
| SZR_OUT_H | 0x1026 | Sizer output height | 0x00f0 | 355 |
| PROC_CTRL_V | 0x0112 | Sizer bypass: bit 4 | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Sizer bypass: bit 4 | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Sizer bypass: bit 4 | 0x0280 | 347 |

## Using the Sizer Example

If the output window from the sensor is the default, VGA landscape (640 x 480) and the wanted output is QQVGA (160 x 120) in video mode and VGA (320 x 240) in still mode, set the SIZE registers as follows:

- Using the SIZE register information on page145, QQVGA for video has a value of 4 for the field VSIZE and VGA for still has a value of 6 for the field SSIZE. Assuming the default values for all other fields of SIZE, the new value is 0x0604.
- To enable the new value, stop the camera module by writing 0x0000 to the CONTROL register (0x0002). Then write 0x0604 to SIZE (0x0008), then configure and start the camera by writing 0x0005 to CONTROL. The camera module is now in video mode, outputting QQVGA size data.

The sizer can be manually controlled for sizes that are not available in the SIZE register. Write the input and output sizes to the video or still sizer registers, then stopping and then re-starting the camera (0x0000 to CONTROL and 0x0001 to CONTROL for video, 0x0002 for still).

### Current Window Size (Read Only)

There are two registers that have the current sensor window size information. The values displayed are the number of row and columns plus the eight extra pixels for the Bayer pattern. These numbers are NOT the sizer output size – these are the sizer input +8. The default values are immediately overwritten on power-up with the current values.

**Sizer input data registers (read only)**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| I_WIDTH | 0x107e | Sizer input width + 8 Bayer pixels | 0x0000 | 376 |
| I_HEIGHT | 0x1080 | Sizer input height + 8 Bayer pixels | 0x0000 | 377 |

## Mirroring / Flipping

The camera module has the capability to mirror the data in both the horizontal and vertical directions, using the sensor. To use mirroring, both the image processor and the sensor must be programmed. If the image processor is programmed for mirroring, it automatically programs the sensor.

Mirroring is controlled with the SFLIP_UD, SFLIP_LF, VFLIP_UD and VFLIP_LR bits in the SIZE register. After configuration, the simple control registers write to the V_VF, V_HF bits in the SEN_CTRL_V register or the S_VF and S_HF bits in the SENS_CTRL_S register.

**NOTE** DO NOT write to bits in the sensor PROC_CTRL_V register, PROC_CTRL_S register or the PROCESS_CTRL register.

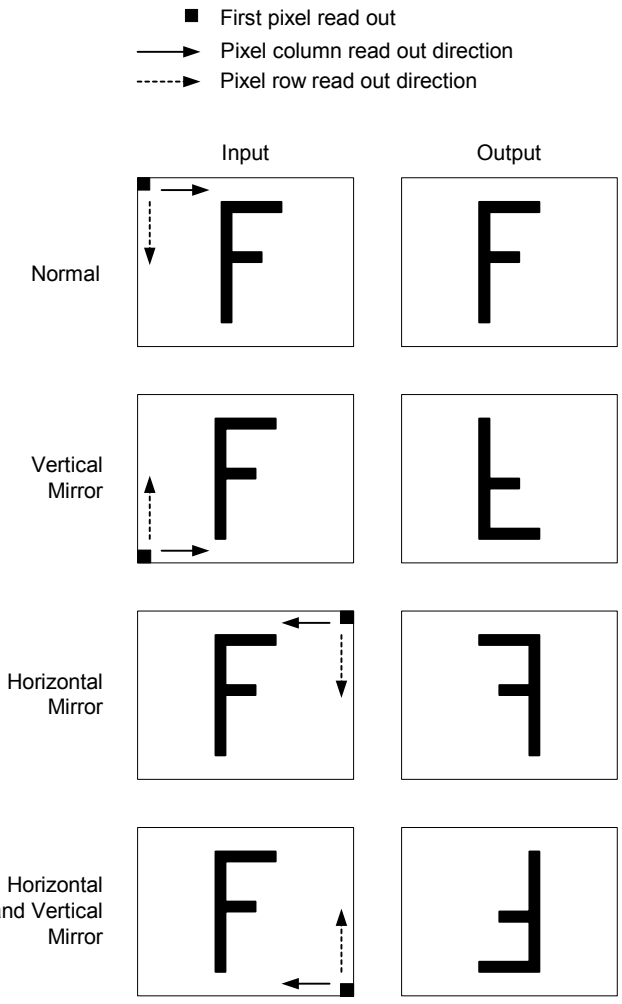



**Figure 8** Mirroring modes

## Auto Exposure and White Balance Functions

The ADCM-2700 camera module offers fully automatic exposure and white balance adjustment. The IC has two types of gain for the pixels; analog gain that is used by the auto exposure function and digital gain that is used by auto white balance.

The auto exposure function adjusts sensor gain and exposure time to meet a programmable target average pixel luminance (AE_TARGET). The auto exposure function uses preferred exposure times that are multiples of twice the line frequency (default is 50 Hz).

Deliberate overexposure can be enabled to keep the camera at a flicker free exposure time under bright lighting conditions. The default is DOE is enabled, but the DOE factor is set to 1.0, or no deliberate overexposure.

The auto white balance function adjusts the sensor's red and blue channel gains relative to the green channel to equalize average pixel luminance. This is the “Gray World” auto white balance. Programmable default, minimum and maximum red/green and blue/green ratios enable control over the auto white balance function. Writing to the APS_COEF_GRN1, APS_COEF_GRN2, APS_COEF_RED and APS_COEF_BLUE registers will manually adjust white balance if auto white balance is disabled.

Flicker detection and correction looks for 50 or 60 Hz flicker in the image and if detected, changes the exposure time to correct for the flicker. The correct value of the master clock must be entered into CLK_PER register.

**Auto functions control registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| AF_CTRL_1 | 0x0140 | Auto functions control 1 | 0x0013 | 220 |
| AF_CTRL_2 | 0x0142 | Auto functions control 2 | 0x0001 | 221 |
| AF_STATUS | 0x0144 | Auto functions status | 0x0000 | 222 |

**Auto exposure registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| AE_DOE_FACTOR | 0x0166 | Deliberate overexposure factor | 0x014e | 239 |
| AE_DOE_MARGIN | 0x0168 | Deliberate overexposure margin | 0x0140 | 240 |
| AE_ETIME_DFLT | 0x015c | Default exposure time | 0x03e8 | 234 |
| AE_ETIME_MAX | 0x015a | Maximum exposure time | 0x4e20 | 233 |
| AE_ETIME_MIN | 0x0158 | Minimum exposure time | 0x0005 | 232 |
| AE_GAIN_DFLT | 0x0156 | Default gain | 0x0200 | 231 |
| AE_GAIN_MAX | 0x0154 | Maximum gain | 0x0500 | 230 |
| AE_GAIN_MIN | 0x0150 | Minimum gain | 0x01c0 | 228 |
| AE_GAIN_MIN_P | 0x0152 | Preferred minimum gain | 0x0200 | 229 |
| AE_MARGIN | 0x0164 | Auto exposure margin | 0x0120 | 238 |
| AE_TARGET | 0x015e | Target auto exposure level | 0x0040 | 235 |
| AE_TOL_ACQ | 0x0160 | Auto exposure tolerance acquire | 0x0118 | 236 |
| AE_TOL_MON | 0x0162 | Auto exposure tolerance monitor | 0x0118 | 237 |
| EREC_PGA | 0x080f | Even row even column (green 1) analog gain | 0x00 | 283 |
| EROC_PGA | 0x0810 | Even row odd column (red) analog gain | 0x00 | 284 |
| OREC_PGA | 0x0811 | Odd row even column (blue) analog gain | 0x00 | 285 |
| OROC_PGA | 0x0812 | Odd row odd column (green 2) analog gain | 0x00 | 286 |

**Auto white balance registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| AWB_RED_MIN | 0x0170 | Minimum red/green ratio | 0x00c0 | 242 |
| AWB_RED_MAX | 0x0172 | Maximum red/green ratio | 0x01a6 | 243 |
| AWB_RED_DFLT | 0x0174 | Default red/green ratio | 0x0134 | 244 |
| AWB_BLUE_MIN | 0x0176 | Minimum blue/green ratio | 0x00c0 | 245 |
| AWB_BLUE_MAX | 0x0178 | Maximum blue/green ratio | 0x02a4 | 246 |
| AWB_BLUE_DFLT | 0x017a | Default blue/green ratio | 0x01e4 | 247 |
| AWB_TOL_ACQ | 0x017c | AWB tolerance acquire | 0x0110 | 248 |
| AWB_TOL_MON | 0x017e | AWB tolerance monitor | 0x0120 | 249 |
| APS_COEF_GRN1 | 0x1062 | Green channel 1 digital gain | 0x0080 | 368 |
| APS_COEF_RED | 0x1064 | Red channel digital gain | 0x0080 | 368 |
| APS_COEF_BLUE | 0x1066 | Blue channel digital gain | 0x0080 | 368 |
| APS_COEF_GRN2 | 0x1068 | Green 2 channel digital gain | 0x0080 | 368 |

## APS_COEF Values

There is one color gain coefficient register for each of the four color components (Green 1, Red, Blue and Green 2). The register values are the initial gain settings if auto white balance is enabled. Whenever auto white balance is disabled, data written to these registers will manually change the image white balance.

The gain values for the APS_COEF registers are stored with three bits used for the integer part of the number and seven bits used for the fractional part of the number. The integer part can range from 0 to 7 and the fractional part has a resolution of one part in 128.

To assemble the gain register, multiply the fractional part of the gain by 128, convert to binary and populate bits 6:0. The integer part is placed in bits 9:7 and bits 15:10 are set to zero. Refer to the following tables for examples.

**Example APS_COEF values**

| Gain | Integer | Integer Binary | Fractional x 128 | Fractional Binary | Reserved Bits | Final Binary | Final Hex |
|---|---|---|---|---|---|---|---|
| 1.0 | 1 | 001 | 0 | 0000000 | 000000 | 000000 001 0000000 | 0x0080 |
| 0.5 | 0 | 000 | 64 | 1000000 | 000000 | 000000 000 1000000 | 0x0040 |
| 0.33 | 0 | 000 | 42 | 0101010 | 000000 | 000000 000 0101010 | 0x002a |
| 2.82 | 2 | 010 | 105 | 1101001 | 000000 | 000000 010 1101001 | 0x0169 |

**Flicker registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CLK_PER | 0x0006 | External clock frequency | 0x32c8 | 144 |
| FLICK_CFG_1 | 0x0182 | Flicker configuration 1 | 0x2aeb | 251 |
| FLICK_CFG_2 | 0x0184 | Flicker configuration 2 | 0x0005 | 252 |

# Anti-Vignetting

The ADCM-2700 has a feature that can correct for the light falloff in the corners of a picture. The anti-vignetting is done by applying different digital gains to each pixel in the picture. The gain is proportional to the square of the distance from the center of the picture. The actual gains used are determined by lookup tables. Each color has it's own lookup table.

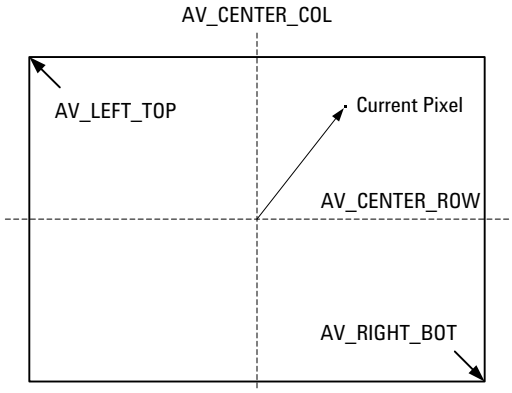



**Figure 9** Anti-vignetting

**Anti-vignetting registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| AV_LEFT_TOP | 0x106a | Upper left window corner:<br>FWROW and FWCOL sensor values | 0x0101 | 369 |
| AV_RIGHT_BOT | 0x106c | Bottom right window corner:<br>LWROW and LWCOL sensor values | 0xa27a | 370 |
| AV_CENTER_COL | 0x106e | Center column of current window:<br>X distance is calculated from this column | 0x0148 | 371 |
| AV_CENTER_ROW | 0x1070 | Center row of current window:<br>Y distance is calculated from this row | 0x00f8 | 372 |
| AV_OVAL_FACT | 0x1094 | Oval factor: < 1 tall/skinny oval, =1 circle, >1 short/fat oval | 0x0100 | 389 |
| AV_OS_GREEN1 | 0x1096 | Green 1 offset: alternate black level / flare subtraction | 0x0000 | 391 |
| AV_OS_RED | 0x1098 | Red offset: alternate black level / flare subtraction | 0x0000 | 391 |
| AV_OS_BLUE | 0x109a | Blue offset: alternate black level/flare subtraction | 0x0000 | 391 |
| AV_OS_GREEN2 | 0x109c | Green 2 offset: alternate black level/flare subtraction | 0x0000 | 391 |

**Anti-vignetting registers (continued)**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PROC_CTRL_V | 0x0112 | Anti-vignetting control: bits 14 and 15 | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Anti-vignetting control: bits 14 and 15 | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Anti-vignetting control: bits 14 and 15 | 0x0280 | 347 |

**Anti-vignetting lookup table base addresses**

| Name | Base Address | Description |
|---|---|---|
| AV_RED | 0x1800 | Anti-vignetting lookup table for red |
| AV_GREEN | 0x1840 | Anti-vignetting lookup table for green |
| AV_BLUE | 0x1880 | Anti-vignetting lookup table for blue |



## Demosaicing

Once the data is read from the sensor, the red, blue and green values need to be determined for each pixel location. The image sensor has a Bayer pattern.

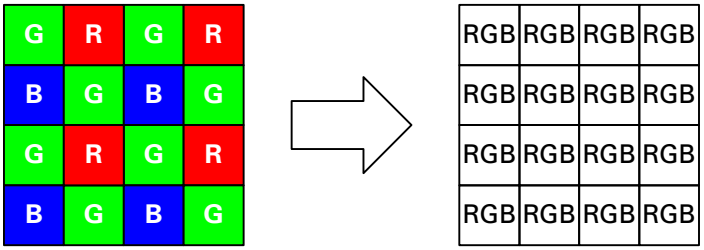



**Figure 10** Demosaic process

The demosaic operation uses information from pixels around each pixel to determine the values of the colors not present at each pixel location.

There are three different demosaic modes:

- **Full demosaic**: data from the pixel and from surrounding pixels are used to determine the red, green and blue values for each pixel.
- **Super pixel demosaic**: data from a cluster of four pixels (two green, a red and a blue) are used to determine the red, green and blue value for a super pixel. A side effect of this mode is that the output size is reduced by half in both directions. If the window is VGA, the super pixel demosaic output will be QVGA.
- **No demosaic** – two output modes available:
  - Data from the current pixel is output on the red channel, with green and blue channels set to zero
  - Data from the current pixel is mapped to the pixel's color filter value and the other two colors are set to zero

**Demosaic registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PROC_CTRL_V | 0x0112 | Bits 2 and 3 control demosaic | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Bits 2 and 3 control demosaic | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Bits 2 and 3 control demosaic | 0x0280 | 347 |

## Sharpening

The ADCM-2700 camera module has the ability to apply a sharpening filter to the image. There are four levels of sharpening: none, low, medium and high (default is to have no sharpening).

**Sharpening registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PROC_CTRL_V | 0x0112 | Bits 11 and 12 control sharpening | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Bits 11 and 12 control sharpening | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Bits 11 and 12 control sharpening | 0x0280 | 347 |

## Halftoning

When outputting data formats with reduced color depth, halftoning is an option to reduce the “banding” that may be present in the image. Halftoning can be set to three values: 4-bit, 6-bit or 8-bit (8-bit actually is halftone off). The default is to have no half toning.

**Halftoning registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PROC_CTRL_V | 0x0112 | Bits 8 and 9 control halftoning | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Bits 8 and 9 control halftoning | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Bits 8 and 9 control halftoning | 0x0280 | 347 |

## Color Correction

The ADCM-2700 camera module uses a programmable color correction function to adjust for the color filter response of the image sensors. Uncorrected RGB values are offset and then multiplied by a 3 x 3 transform coefficient matrix.

$$\begin{bmatrix} R_{Corrected} \\ G_{Corrected} \\ B_{Corrected} \end{bmatrix} = \begin{bmatrix} Pre\_offset\_0 \\ Pre\_offset\_1 \\ Pre\_offset\_2 \end{bmatrix} + \begin{bmatrix} CC\_00 & CC\_01 & CC\_02 \\ CC\_10 & CC\_11 & CC\_12 \\ CC\_20 & CC\_21 & CC\_22 \end{bmatrix} \times \left( \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Post\_offset\_0 \\ Post\_offset\_1 \\ Post\_offset\_2 \end{bmatrix} \right)$$

The default settings are for an RGB sensor and fluorescent light:

$$\begin{bmatrix} R_{Corrected} \\ G_{Corrected} \\ B_{Corrected} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 2.2070 & -0.7266 & -0.4805 \\ -0.6094 & 1.9844 & -0.3789 \\ 0.0313 & -1.4492 & 2.4180 \end{bmatrix} \times \left( \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} -8 \\ -8 \\ -8 \end{bmatrix} \right)$$

The color correction registers also allow for different kinds of illuminants.

**Other Illuminants**

| D65 Illuminant | Tungsten Illuminant |
|---|---|
| $\begin{bmatrix} 2.2070 & -0.4414 & -0.7695 \\ -0.6133 & 1.9375 & -0.3242 \\ 0.0508 & -1.3047 & 2.2539 \end{bmatrix}$ | $\begin{bmatrix} 2.2266 & -0.7969 & -0.4297 \\ -0.6484 & 2.0938 & -0.4453 \\ -0.0234 & -2.0820 & 3.1055 \end{bmatrix}$ |

**Notes:**

- Each row within each matrix should add up to the same number
- D65 is the CIE illuminant for 6500K and represents sunlight
- Tungsten is a standard light bulb

## Color Correction Coefficient Registers

Each color correction coefficient is stored as a 12-bit signed 2's complement number, with four bits used for the signed integer part of the number and eight bits used for the fractional part of the number. The coefficients and their registers are named by the coefficient's row and column position in the 3 x 3 color correction matrix. The numbers can range from -8.000 to +7.996.

If the number is positive, multiply it by 256 and convert to hex. If negative, multiply by 256, add to 4096 and convert to hex. These operations will also set the reserved bits to 0. The value of the reserved bits does not matter. Refer to the following tables for examples.

**Color correction coefficient register number format**

| Positive Numbers | | | Negative Numbers | | | |
|---|---|---|---|---|---|---|
| Matrix Number | x 256 | Final Hex | Matrix Number | x 256 | 4096 + N | Final Hex |
| 0 | 0 | 0x0000 | - 0.004 | - 1 | 4095 | 0x0fff |
| 0.004 | 1 | 0x0001 | - 0.008 | - 2 | 4094 | 0x0ffe |
| – | – | – | – | – | – | – |
| 7.992 | 2046 | 0x07fe | - 7.996 | - 2047 | 2049 | 0x0801 |
| 7.996 | 2047 | 0x07ff | - 8.000 | - 2048 | 2048 | 0x0800 |

**Color correction coefficient registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CC_COEF_00 | 0x1028 | Color correction coefficient 00 | 0x02f9 | 358 |
| CC_COEF_01 | 0x102a | Color correction coefficient 01 | 0x0f03 | 358 |
| CC_COEF_02 | 0x102c | Color correction coefficient 02 | 0x0f02 | 358 |
| CC_COEF_10 | 0x102e | Color correction coefficient 10 | 0x0f4f | 358 |
| CC_COEF_11 | 0x1030 | Color correction coefficient 11 | 0x025c | 358 |
| CC_COEF_12 | 0x1032 | Color correction coefficient 12 | 0x0f54 | 358 |
| CC_COEF_20 | 0x1034 | Color correction coefficient 20 | 0x0fe0 | 358 |
| CC_COEF_21 | 0x1036 | Color correction coefficient 21 | 0x0e4a | 358 |
| CC_COEF_22 | 0x1038 | Color correction coefficient 22 | 0x02d5 | 358 |

If other illuminants are wanted and auto white balance is turned off, the values from the matrices above need to be converted into the signed 2's complement numbers for the registers. The following table shows the hex coefficient values and their hex addresses. These values are not the defaults, but result in more saturated colors.

**Illuminant values to use**

| | Address | 0x1028 | 0x102a | 0x102c | 0x102e | 0x1030 | 0x1032 | 0x1034 | 0x1036 | 0x1038 |
|---|---|---|---|---|---|---|---|---|---|---|
| Illuminant | D65 | 027b | 0f52 | 0f32 | 0f3f | 0262 | 0f5b | 0ff0 | 0e99 | 0276 |
| | Fluorescent | 026f | 0f07 | 0f88 | 0f3e | 026c | 0f54 | 0fee | 0e64 | 02ad |
| | Tungsten | 027f | 0fd2 | 0fad | 0f2c | 029e | 0f35 | 0fc2 | 0db0 | 038d |

## Color Correction Offset Registers

The color correction offset registers can be used to correct for flare or scattered light, or to bring the black values back to wanted levels. Negative numbers reduce scattered light and lower the black levels.

Each color correction offset is stored in a 9-bit read / write register. The numbers are 2's complement. Since only nine bits are used, the numbers can range from -256 to +255.

If the number is positive, convert it to hex. If negative, add it to 512 and convert it to hex. These operations will also set the reserved bits to 0. The value of the reserved bits does not matter. Refer to the following table for examples.

**Color correction offset register number format**

| Positive Numbers | | Negative Numbers | |
|---|---|---|---|
| **Offset Number** | **Final Hex** | **Matrix Number** | **Final Hex** |
| 0 | 0x0000 | - 1 | 0x01ff |
| 1 | 0x0001 | - 2 | 0x01fe |
| – | – | – | – |
| 254 | 0x00fe | - 255 | 0x0001 |
| 255 | 0x00ff | - 256 | 0x0000 |

The offsets and their registers are named by the offset's row position in the 3 x 1 offset matrix. If the default value results in a washed out looking picture, try using -8 (decimal), 0x01f8 (hex).

**Color correction offset registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CC_PRE_OS_0 | 0x103a | Color correction pre offset 0 | 0x01f8 | 360 |
| CC_PRE_OS_1 | 0x103c | Color correction pre offset 1 | 0x01f8 | 360 |
| CC_PRE_OS_2 | 0x103e | Color correction pre offset 2 | 0x01f8 | 360 |
| CC_POST_OS_0 | 0x1040 | Color correction post offset 0 | 0x0000 | 360 |
| CC_POST_OS_1 | 0x1042 | Color correction post offset 1 | 0x0000 | 360 |
| CC_POST_OS_2 | 0x1044 | Color correction post offset 2 | 0x0000 | 360 |

## Noise Adaptive Color Correction

The Noise Adaptive Color Correction (NACC) adjusts the color correction coefficients based on the light level in a scene. The brighter the light, the more saturated the colors. The NACC uses the Exposure/Gain Product (EGP) to estimate the light level. The higher the EGP, the dimmer the light.

The lookup table can have up to 8 entries, each consisting of an EGP and a saturation level. The saturation level is a number from 0 to 1 and is saved in 8.8 format. A saturation level is selected when the current EGP is less than or equal to an EGP in the table. EGP values must be in descending order. The EGP values in the table are divided by a scale factor of 16 to allow for a greater range.

**NACC control register**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| AF_CTRL1 | 0x0140 | NACC controlled by bit 3 | 0x0013 | 220 |

**NACC table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_EGP_1 | 0x0250 | 0x04 | 0x50 | 0x05dc | EGP = 6000 x 4/16 = 1500 = 0x05dc |
| NACC_SAT_1 | 0x0252 | 0x04 | 0x52 | 0x0000 | Saturation = 0.0, 0% |
| NACC_EGP_2 | 0x0254 | 0x04 | 0x54 | 0x0465 | EGP = 6000 x 3/16 = 1150 = 0x0465 |
| NACC_SAT_2 | 0x0256 | 0x04 | 0x56 | 0x0040 | Saturation = 0.25, 25% |
| NACC_EGP_3 | 0x0258 | 0x04 | 0x58 | 0x02ee | EGP = 6000 x 2/16 = 750 = 0x02ee |
| NACC_SAT_3 | 0x025a | 0x04 | 0x5a | 0x0080 | Saturation = 0.5, 50% |
| NACC_EGP_4 | 0x025c | 0x04 | 0x5c | 0x0177 | EGP = 6000 x 1/16 = 375 = 0x0177 |
| NACC_SAT_4 | 0x025e | 0x04 | 0x5e | 0x00c0 | Saturation = 0.75, 75% |
| NACC_EGP_5 | 0x0260 | 0x04 | 0x60 | 0x0000 | EGP = 0 = 00 = 0x0000 |
| NACC_SAT_5 | 0x0262 | 0x04 | 0x62 | 0x0100 | Saturation = 1.0, 100% |
| NACC_EGP_6 | 0x0264 | 0x04 | 0x64 | 0x0000 | |
| NACC_SAT_6 | 0x0266 | 0x04 | 0x66 | 0x0000 | |
| NACC_EGP_7 | 0x0268 | 0x04 | 0x68 | 0x0000 | |
| NACC_SAT_7 | 0x026a | 0x04 | 0x6a | 0x0000 | |
| NACC_EGP_8 | 0x026c | 0x04 | 0x6c | 0x0000 | |
| NACC_SAT_8 | 0x026e | 0x04 | 0x6e | 0x0000 | |

The noise adaptive color correction needs to know the limits of the color correction matrices.

## NACC Bright and Dark Tables

There are two tables: the bright coefficients table (which is the normal color corrections coefficients table) and the dark coefficients table (which corresponds to a grayscale table). Depending on the value in the NACC table, the CC_COEF_XX values are updated with an interpolated value between the bright and dark tables.

When NACC is enabled, the simple control register loads the current CC_COEF_XX values into the NACC bright coefficients table. When NACC is disabled, the bright coefficients table is written out to the CC_COEF table. If the default color correction table needs to be changed, the proper order is to turn off NACC, change the color correction table and then re-enable NACC.

**NACC bright coefficients table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_BC_00 | 0x0270 | 0x04 | 0x70 | 0x0235 | Value for CC_COEF_00 |
| NACC_BC_01 | 0x0272 | 0x04 | 0x72 | 0xff46 | Value for CC_COEF_01 |
| NACC_BC_02 | 0x0274 | 0x04 | 0x74 | 0xff85 | Value for CC_COEF_02 |
| NACC_BC_10 | 0x0276 | 0x04 | 0x76 | 0xff64 | Value for CC_COEF_10 |
| NACC_BC_11 | 0x0278 | 0x04 | 0x78 | 0x01fc | Value for CC_COEF_11 |
| NACC_BC_12 | 0x027a | 0x04 | 0x7a | 0xff9f | Value for CC_COEF_12 |
| NACC_BC_20 | 0x027c | 0x04 | 0x7c | 0x0008 | Value for CC_COEF_20 |
| NACC_BC_21 | 0x027e | 0x04 | 0x7e | 0xfe8d | Value for CC_COEF_21 |
| NACC_BC_22 | 0x0280 | 0x05 | 0x00 | 0x026b | Value for CC_COEF_22 |

**NACC dark coefficients table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_DC_00 | 0x0282 | 0x05 | 0x02 | 0x0048 | Value for CC_COEF_00 |
| NACC_DC_01 | 0x0284 | 0x05 | 0x04 | 0x010b | Value for CC_COEF_01 |
| NACC_DC_02 | 0x0286 | 0x05 | 0x06 | 0xffaa | Value for CC_COEF_02 |
| NACC_DC_10 | 0x0288 | 0x05 | 0x08 | 0x0048 | Value for CC_COEF_10 |
| NACC_DC_11 | 0x028a | 0x05 | 0x0a | 0x010b | Value for CC_COEF_11 |
| NACC_DC_12 | 0x028c | 0x05 | 0x0c | 0xffaa | Value for CC_COEF_12 |
| NACC_DC_20 | 0x028e | 0x05 | 0x0e | 0x0048 | Value for CC_COEF_20 |
| NACC_DC_21 | 0x0290 | 0x05 | 0x10 | 0x010b | Value for CC_COEF_21 |
| NACC_DC_22 | 0x0292 | 0x05 | 0x12 | 0xffaa | Value for CC_COEF_22 |

## Gamma Correction (Tone Mapping)

The ADCM-2700 camera module provides two lookup tables for non-linearly transforming red, green and blue pixel intensities to compensate for display device characteristics. One is for video mode and the other is for still mode.

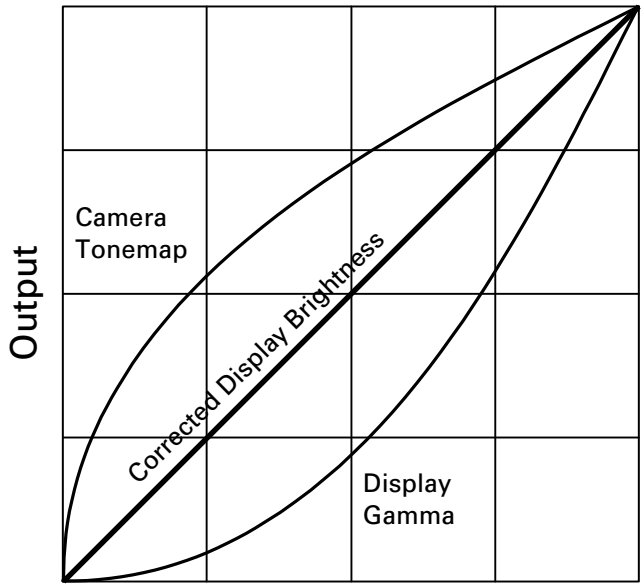



**Figure 11** Gamma curve

Each lookup table can be linear or bottom-weighted. The linear lookup table divides the input intensity range into equal intervals while the bottom-weighted lookup table divides the input range into increasingly larger intervals, starting with the smallest interval at the bottom end of the intensity range. Both types of lookup tables contain 33 entries and each color (red, green and blue) has a separate table.

During gamma correction, the ADCM-2700 camera module tries to match the input pixel's intensity to an input level in the lookup table. If there is an exact match, the corresponding output value is used. If there is no match, the module linearly interpolates between two table entries to determine an output value for a given input pixel intensity. Because the bottom-weighted lookup table has more entries at the lower end of the input range, it describes the transfer function for low-intensity (dark) pixels more precisely than the linear lookup table.

There are separate sets of lookup tables for video and still modes. In its initial (reset) configuration state, the module uses the bottom-weighted lookup table. See Chapter D, “Tonemaps (Gamma),” starting on page 461 for default values (the default tonemap is sRGB).

**Tonemap enable and weighting registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PROC_CTRL_V | 0x0112 | Bits 6 and 7 enable tonemap and set weighting | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | Bits 6 and 7 enable tonemap and set weighting | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | Bits 6 and 7 enable tonemap and set weighting | 0x0280 | 347 |

**Tonemap base addresses**

| Name | Base Address | Description |
|---|---|---|
| Video “All” tonemap | 0x01c0 | Tonemap loaded into RAM when RUN bit is set |
| Still “All” tonemap | 0x0202 | Tonemap loaded into RAM when SNAP bit is set |
| “All” tonemap | 0x1400 | RAM tonemap used for all colors |
| Red tonemap | 0x1480 | RAM tonemap for red. Autoloaded from “All” tonemap |
| Green tonemap | 0x1500 | RAM tonemap for green; autoloaded from “All” tonemap |
| Blue tonemap | 0x1580 | RAM tonemap for blue; autoloaded from “All” tonemap |

## Lookup Table Addresses

The address of a table output entry equals the lookup tables base address + (2 x table entry number).

If the same gamma correction is used for all three colors, write one set of table entries using the “All” base address. The ADCM-2700 camera module copies the “All” table entries to the red, green and blue lookup tables.

## Using Custom Tonemaps

The look up tables can be replaced with custom tonemaps. This is usually done to implement special effects such as negative and solarization. The input range (inputs are 10-bit values) is fixed to either the linear or bottom-weighted values (see “Linear and Bottom-Weighted Inputs" on page 462 for values). Output values for table entries 0-31 are 10-bit values, while table entry 32 is an 11-bit value. Table entries can range from 0 to 1024. If the non-default output values are on a 0.0 to 1.0 range, multiply them by 1024 to compute the value to load into the table.

# Color Space Conversion

The ADCM-2700 camera module provides a programmable color space conversion function to convert RGB values to other color spaces. RGB values are multiplied by a 3 x 3 transform coefficient matrix and then offset:

$$\begin{bmatrix} \text{New Color Space 0} \\ \text{New Color Space 1} \\ \text{New Color Space 2} \end{bmatrix} = \begin{bmatrix} \text{CSC\_00} & \text{CSC\_01} & \text{CSC\_02} \\ \text{CSC\_10} & \text{CSC\_11} & \text{CSC\_12} \\ \text{CSC\_20} & \text{CSC\_21} & \text{CSC\_22} \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} \text{Offset\_0} \\ \text{Offset\_1} \\ \text{Offset\_2} \end{bmatrix}$$

## RGB to YCbCr

The default conversion is from RGB to YCbCr. The default coefficients for this conversion are:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299000 & 0.587000 & 0.144000 \\ -0.168736 & -0.331264 & 0.500000 \\ -0.500000 & -0.418688 & -0.081312 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

However, due to the number of bits stored in the camera module, the actual matrix used is:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.2968750 & 0.5859375 & 0.1171875 \\ -0.1718750 & -0.3281250 & 0.5000000 \\ -0.5000000 & -0.4218750 & -0.0781250 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

## RGB to YUV

Another conversion is from RGB to YUV. The coefficients for this are:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Due to the number of bits stored in the camera module, the actual matrix used is:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.2968750 & 0.5859375 & 0.1171875 \\ -0.1484375 & -0.2890625 & 0.4375000 \\ 0.6171875 & -0.5156250 & -0.10156250 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

## Raw Data – RGB Output

Another conversion is from RGB to RGB, with no conversion. The coefficients for this conversion are:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

## Color Space Conversion Coefficient Registers

The color space conversion matrix values are stored as a 9-bit signed 2's complement number, with two bits used for the signed integer part of the number and seven bits used for the fractional part of the number. The numbers can range from - 2.000 to +1.996. If the number is positive, multiply it by 128 and convert to hex. If negative, multiply by 128, add to 512 and convert to hex.

These operations will also set the reserved bits to 0. The value of the reserved bits does not matter. There are separate tables for video and still modes. When a mode is selected, it is copied to the working registers. Refer to the following tables for examples.

**Color space conversion coefficient number**

| Positive Numbers | | | Negative Numbers | | | |
|---|---|---|---|---|---|---|
| **Number** | **x 128** | **Final Hex** | **Number** | **x 128** | **512 + N** | **Final Hex** |
| 0 | 0 | 0x0000 | - 0.008 | - 1 | 511 | 0x01ff |
| 0.008 | 1 | 0x0001 | - 0.016 | - 2 | 510 | 0x01fe |
| . . . | . . . | . . . | . . . | . . . | . . . | . . . |
| 1.988 | 254 | 0x00fe | - 1.992 | - 255 | 257 | 0x0101 |
| 1.996 | 255 | 0x00ff | - 2.000 | - 256 | 256 | 0x0100 |

The coefficients and their registers are named by the coefficient's row and column position in the color space conversion matrix.

**Color space conversion registers, video mode**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CSC_00_V | 0x0190 | Color space conversion coefficient 00 | 0x0026 | 256 |
| CSC_01_V | 0x0192 | Color space conversion coefficient 01 | 0x004b | 256 |
| CSC_02_V | 0x0194 | Color space conversion coefficient 02 | 0x000f | 256 |
| CSC_10_V | 0x0196 | Color space conversion coefficient 10 | 0x01ed | 256 |
| CSC_11_V | 0x0198 | Color space conversion coefficient 11 | 0x01db | 256 |
| CSC_12_V | 0x019a | Color space conversion coefficient 12 | 0x0038 | 256 |
| CSC_20_V | 0x019c | Color space conversion coefficient 20 | 0x004f | 256 |
| CSC_21_V | 0x019e | Color space conversion coefficient 21 | 0x01be | 256 |
| CSC_22_V | 0x01a0 | Color space conversion coefficient 22 | 0x01f3 | 256 |

**Color space conversion registers, still mode**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CSC_00_S | 0x01a8 | Color space conversion coefficient 00 | 0x0026 | 258 |
| CSC_01_S | 0x01aa | Color space conversion coefficient 01 | 0x004b | 258 |
| CSC_02_S | 0x01ac | Color space conversion coefficient 02 | 0x000f | 258 |
| CSC_10_S | 0x01ae | Color space conversion coefficient 10 | 0x01ed | 258 |
| CSC_11_S | 0x01b0 | Color space conversion coefficient 11 | 0x01db | 258 |
| CSC_12_S | 0x01b2 | Color space conversion coefficient 12 | 0x0038 | 258 |
| CSC_20_S | 0x01b4 | Color space conversion coefficient 20 | 0x004f | 258 |
| CSC_21_S | 0x01b6 | Color space conversion coefficient 21 | 0x01be | 258 |
| CSC_22_S | 0x0188 | Color space conversion coefficient 22 | 0x01f3 | 258 |

**Color space conversion registers, currently selected mode**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CSC_COEF_00 | 0x1046 | Color space conversion coefficient 00 | 0x0026 | 363 |
| CSC_COEF_01 | 0x1048 | Color space conversion coefficient 01 | 0x004b | 363 |
| CSC_COEF_02 | 0x104a | Color space conversion coefficient 02 | 0x000f | 363 |
| CSC_COEF_10 | 0x104c | Color space conversion coefficient 10 | 0x01ed | 363 |
| CSC_COEF_11 | 0x104e | Color space conversion coefficient 11 | 0x01db | 363 |
| CSC_COEF_12 | 0x1050 | Color space conversion coefficient 12 | 0x0038 | 363 |
| CSC_COEF_20 | 0x1052 | Color space conversion coefficient 20 | 0x004f | 363 |
| CSC_COEF_21 | 0x1054 | Color space conversion coefficient 21 | 0x01be | 363 |
| CSC_COEF_22 | 0x1056 | Color space conversion coefficient 22 | 0x01f3 | 363 |

**Color space conversion register values, different output formats**

| Register name | Address | Description | YCbCr | YUV | Raw Data |
|---|---|---|---|---|---|
| CSC_00_V | 0x0190 | CSC coefficient 00 – video | 0x0026 | 0x0026 | 0x0080 |
| CSC_01_V | 0x0192 | CSC coefficient 01 – video | 0x004b | 0x004b | 0x0080 |
| CSC_02_V | 0x0194 | CSC coefficient 02 – video | 0x000f | 0x000f | 0x0080 |
| CSC_10_V | 0x0196 | CSC coefficient 10 – video | 0x01ea | 0x01ed | 0x0000 |
| CSC_11_V | 0x0198 | CSC coefficient 11 – video | 0x01d6 | 0x01db | 0x0000 |
| CSC_12_V | 0x019a | CSC coefficient 12 – video | 0x0040 | 0x0038 | 0x0000 |
| CSC_20_V | 0x019c | CSC coefficient 20 – video | 0x0040 | 0x004f | 0x0000 |
| CSC_21_V | 0x019e | CSC coefficient 21 – video | 0x01ca | 0x01be | 0x0000 |
| CSC_22_V | 0x01a0 | CSC coefficient 22 – video | 0x01f6 | 0x01f3 | 0x0000 |
| CSC_00_S | 0x01a8 | CSC coefficient 00 – still | 0x0026 | 0x0026 | 0x0080 |
| CSC_01_S | 0x01aa | CSC coefficient 01 – still | 0x004b | 0x004b | 0x0080 |
| CSC_02_S | 0x01ac | CSC coefficient 02 – still | 0x000f | 0x000f | 0x0080 |
| CSC_10_S | 0x01ae | CSC coefficient 10 – still | 0x01ea | 0x01ed | 0x0000 |
| CSC_11_S | 0x01b0 | CSC coefficient 11 – still | 0x01d6 | 0x01db | 0x0000 |
| CSC_12_S | 0x01b2 | CSC coefficient 12 – still | 0x0040 | 0x0038 | 0x0000 |
| CSC_20_S | 0x01b4 | CSC coefficient 20 – still | 0x0040 | 0x004f | 0x0000 |
| CSC_21_S | 0x01b6 | CSC coefficient 21 – still | 0x01ca | 0x01be | 0x0000 |
| CSC_22_S | 0x0188 | CSC coefficient 22 – still | 0x01f6 | 0x01f3 | 0x0000 |
| CSC_COEF_00 | 0x1046 | CSC coefficient 00 – current | 0x0026 | 0x0026 | 0x0080 |
| CSC_COEF_01 | 0x1048 | CSC coefficient 01 – current | 0x004b | 0x004b | 0x0080 |
| CSC_COEF_02 | 0x104a | CSC coefficient 02 – current | 0x000f | 0x000f | 0x0080 |
| CSC_COEF_10 | 0x104c | CSC coefficient 10 – current | 0x01ea | 0x01ed | 0x0000 |
| CSC_COEF_11 | 0x104e | CSC coefficient 11 – current | 0x01d6 | 0x01db | 0x0000 |
| CSC_COEF_12 | 0x1050 | CSC coefficient 12 – current | 0x0040 | 0x0038 | 0x0000 |
| CSC_COEF_20 | 0x1052 | CSC coefficient 20 – current | 0x0040 | 0x004f | 0x0000 |
| CSC_COEF_21 | 0x1054 | CSC coefficient 21 – current | 0x01ca | 0x01be | 0x0000 |
| CSC_COEF_22 | 0x1056 | CSC coefficient 22 – current | 0x01f6 | 0x01f3 | 0x0000 |

## Color Space Conversion Offset Registers

The YCbCr color space requires that the Cb and Cr data be offset by +128. The color space conversion offset registers are used to provide this offset. The numbers are nine bits long, always positive, without fractional components. The range is from 0 to 512. The offsets and their registers are named by the offset's row position in the 3 x 1 offset matrix.

**Color space conversion offset registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CSC_OS0_V | 0x01a2 | Color space conversion offset 0, video | 0x0000 | 257 |
| CSC_OS1_V | 0x01a4 | Color space conversion offset 1, video | 0x0080 | 257 |
| CSC_OS2_V | 0x01a6 | Color space conversion offset 2 , video | 0x0080 | 257 |
| CSC_OS0_S | 0x01ba | Color space conversion offset 0, still | 0x0000 | 259 |
| CSC_OS1_S | 0x01bc | Color space conversion offset 1, still | 0x0080 | 259 |
| CSC_OS2_S | 0x01be | Color space conversion offset 2, still | 0x0080 | 259 |
| CSC_OS_0 | 0x1058 | Color space conversion offset 0, working | 0x0000 | 364 |
| CSC_OS_1 | 0x105a | Color space conversion offset 1, working | 0x0080 | 364 |
| CSC_OS_2 | 0x105c | Color space conversion offset 2, working | 0x0080 | 364 |

**Color space conversion offset register values, different output formats**

| Register name | Address | Description | YCbCr | YUV | Raw Data |
|---|---|---|---|---|---|
| CSC_OS0_V | 0x01a2 | Video CSC offsets | 0x0000 | 0x0000 | 0x0000 |
| CSC_OS1_V | 0x01a4 | Video CSC offsets | 0x0080 | 0x0000 | 0x0000 |
| CSC_OS2_V | 0x01a6 | Video CSC offsets | 0x0080 | 0x0000 | 0x0000 |
| CSC_OS0_S | 0x01ba | Still CSC offsets | 0x0000 | 0x0000 | 0x0000 |
| CSC_OS1_S | 0x01bc | Still CSC offsets | 0x0080 | 0x0000 | 0x0000 |
| CSC_OS2_S | 0x01be | Still CSC offsets | 0x0080 | 0x0000 | 0x0000 |
| CSC_OS_0 | 0x1058 | Working CSC offsets | 0x0000 | 0x0000 | 0x0000 |
| CSC_OS_1 | 0x105a | Working CSC offsets | 0x0080 | 0x0000 | 0x0000 |
| CSC_OS_2 | 0x105c | Working CSC offsets | 0x0080 | 0x0000 | 0x0000 |

# Image Statistics and Histograms

This section explains image statistics and histograms.

## Image Statistics

The ADCM-2700 camera module provides a suite of statistics registers. These registers provide peak pixel intensity, pixel sums and the number of pixel values clipped to minimum or maximum values to monitor and adjust the camera module for specialized conditions. These statistics are gathered over a 4x super pixel; the 2 x 2 pixels of green 1, red, blue and green 2.

**Image statistics registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| STAT_CAP_CTRL | 0x1072 | Statistics initialization | 0x0021 | 373 |
| STAT_MODE_CTRL | 0x1074 | Statistics configuration | 0x0000 | 374 |
| GREEN1_SUM | 0x1076 | Green 1 pixels sum | 0x0000 | 375 |
| GREEN2_SUM | 0x107c | Green 2 pixels sum | 0x0000 | 375 |
| RED_SUM | 0x1078 | Red pixels sum | 0x0000 | 375 |
| BLUE_SUM | 0x107a | Blue pixels sum | 0x0000 | 375 |

## Image Histograms

The ADCM-2700's image statistics function includes the generation of image histograms. Triggered by the GO field of the STAT_CAP_CTRL, the camera module computes a 20-bin histogram for each pixel type (green 1, red, blue and green 2) using a 10-bit intensity range. Bins are 64 units wide with the exception of the bins 0 through 2 and bins 17 through 19, which are narrower. This provides more detailed information at the limits of the image sensor.

Bins 0, 1, 18 and 19 are 16 units in width and width and bins 2 and 17 are 32 units in width. Each bin contains 1/4 number of pixels falling within the bin's pixel value range. The pixels are averaged using a 2x2 boxcar filter before being sorted into bins. The FCNT field of the STAT_CAP_CTRL determines how many frames are used for the histogram generation. The maximum count for a bin is 0x0fff and pixels that fall into a bin after the maximum is reached are not counted (e.g., the count stays at 0x0fff).

### Image Statistics RAM

Addresses as shown below. Bin counts are 12-bit values and bin numbers range from 0 to 19.

**Image histogram registers**

| Histogram | Base Address | Bin Addresses | Default | Page |
|---|---|---|---|---|
| Green 1 | 0x1600 | base + (2 x bin number) | 0x0000 | 451 |
| Red | 0x1028 | base + (2 x bin number) | 0x0000 | 452 |
| Blue | 0x1050 | base + (2 x bin number) | 0x0000 | 453 |
| Green 2 | 0x1078 | base + (2 x bin number) | 0x0000 | 454 |

**Histogram format**

| Histogram Bin | Width | Pixel Value Range | Histogram Bin | Width | Pixel Value Range |
|---|---|---|---|---|---|
| 0 | 16 | 0 – 15 | 10 | 64 | 512 – 575 |
| 1 | 16 | 16 – 31 | 11 | 64 | 576 – 639 |
| 2 | 32 | 32 – 63 | 12 | 64 | 640 – 703 |
| 3 | 64 | 64 – 127 | 13 | 64 | 704 – 767 |
| 4 | 64 | 128 – 191 | 14 | 64 | 768 – 831 |
| 5 | 64 | 192 – 255 | 15 | 64 | 832 – 895 |
| 6 | 64 | 256 – 319 | 16 | 64 | 896 – 959 |
| 7 | 64 | 320 – 383 | 17 | 32 | 960 – 991 |
| 8 | 64 | 384 – 447 | 18 | 16 | 992 – 1007 |
| 9 | 64 | 448 – 511 | 19 | 16 | 1008 – 1023 |



## Bin Number vs. Pixel Values

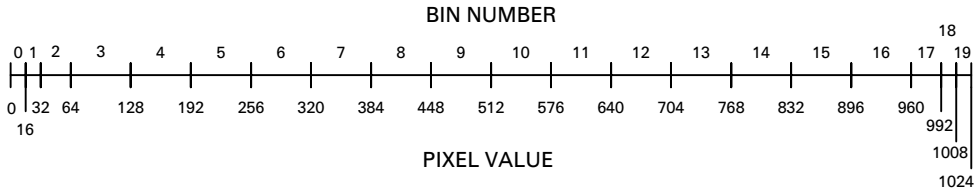



**Figure 12** Bin number vs. pixel values

## Flicker Statistics

When auto flicker is enabled, the histogram section of RAM is used to determine if the current image has flicker present. To do this, the statistics block adds up all of the green pixels on certain rows in two consecutive frames and stores the sums for each row in the flicker RAM. The sums are 18 bits, with the upper 16 stored in the RAM.

The algorithm always stores the data for the first row of each frame, then skips N rows, accumulates the next row, skips N rows, etc. N is the flicker skip count, which is the upper four bits of STAT_MODE_CTRL, address 0x1074.

The flicker statistics block accumulates a maximum of 48 row sums per frame. Since the auto flicker algorithm needs data from two frames, there is a maximum of 96 row sums. If the statistics block accumulates the 48th row sum before it reaches the end of the frame, it will stop accumulating until the first row of the next frame.

If the statistics block reaches the end of the frame before accumulating the 48th row, it stores the row sum for the first row of the second frame in the next available RAM location. This means that depending on the number of rows in the frame and the flicker skip count, there may be less than 48 row sums. You need to know this to be able to interpret the flicker statistics RAM data.

The following shows the number of row sums for a particular frame size and flicker skip count:

$$\text{Number of Row Sums} = \min\left[48, \text{floor}\left(\frac{\text{ImageHeight} - 1}{\text{SkipCount} + 1}\right) + 1\right]$$

Since the auto flicker algorithm operates on the data from the sensor before any image processing, the ImageHeight parameter includes the sacrificial Bayer pixels. The following table shows the number of row sums for various flicker skip counts and for the CIF and QVGA window sizes.

**Number of row sums**

| Flicker Skip Count | VGA | QVGA | CIF | QCIF | Flicker Skip Count | VGA | QVGA | CIF | QCIF |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 48 | 48 | 48 | 48 | 8 | 48 | 28 | 33 | 17 |
| 1 | 48 | 48 | 48 | 48 | 9 | 48 | 25 | 30 | 16 |
| 2 | 48 | 48 | 48 | 48 | 10 | 45 | 23 | 27 | 14 |
| 3 | 48 | 48 | 48 | 38 | 11 | 41 | 21 | 25 | 13 |
| 4 | 48 | 48 | 48 | 31 | 12 | 38 | 20 | 23 | 12 |
| 5 | 48 | 42 | 48 | 26 | 13 | 35 | 18 | 22 | 11 |
| 6 | 48 | 36 | 43 | 22 | 14 | 33 | 17 | 20 | 11 |
| 7 | 48 | 31 | 37 | 19 | 15 | 31 | 16 | 19 | 10 |

## CCIR 656 Data

The data is sent using an 8-bit parallel, CCIR 656 interface, with the option of both external VSYNC and HSYNC, or embedded synchronization codes (see Chapter 5, “Data Output Formats,” starting on page 105 and Chapter 4, “Parallel Data Output Protocols,” starting on page 83 for full details on the interface). To meet the CCIR 656 timing specifications, the VCLK clock must be 27 MHz.

**CCIR 656 registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CCIR_TIMING | 0x1010 | CCIR interface timing | 0x0000 | 343 |
| CCIR_TIMING2 | 0x1086 | CCIR interface timing | 0x0000 | 381 |
| CCIR_TIMING3 | 0x1088 | CCIR interface timing | 0x0010 | 382 |
| HSYNC_PER_V | 0x0116 | Horizontal synchronization period – video mode | 0x0a8b | 197 |
| HSYNC_PER_S | 0x0136 | Horizontal synchronization period – still mode | 0x0545 | 215 |
| HSYNC_PER | 0x1060 | Horizontal synchronization period | 0x0a8b | 366 |
| R_Y_MAX_MIN | 0x1012 | Luminance (or red) maximum and minimum | 0xff00 | 344 |
| G_CB_MAX_MIN | 0x1014 | Chrominance Cb (or green) maximum and minimum | 0xff00 | 345 |
| B_CR_MAX_MIN | 0x1016 | Chrominance Cr (or blue) maximum and minimum | 0xff00 | 346 |

## Serial Control Interface

The camera module is controlled using a 2-wire serial interface, which is used to read and write to the registers in the image sensor and image processor. See Chapter 3, “Serial Control Interface,” starting on page 67 for additional information.

# Bad Pixel Control

If a pixel value deviates sufficiently from the local average, it is considered a bad pixel. If the image has a lot of sharp edges with high contrast at high frequency (such as close up pictures of business cards), the BPA algorithm may introduce undesirable artifacts. Under these conditions, a better picture may be obtained by turning BPA off. The state of BPA correction is controlled in the BYPASS_CTRL register.

**Bad pixel registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| BPA_SF_GTHRESH | 0x101a | BPA scale factor | 0x0220 | 349 |
| BPA_OUTL_PED | 0x101c | BPA outlier and pedestal value | 0x4008 | 350 |
| BPA_BADPIX_CNT | 0x101e | Number of bad pixels in the current frame | 0x0000 | 351 |
| BPA_D2_THRESH | 0x108c | BPA second derivative threshold | 0x0100 | 384 |
| PROC_CTRL_V | 0x0112 | BPA bypass | 0x0280 | 194 |
| PROC_CTRL_S | 0x0132 | BPA bypass | 0x0280 | 212 |
| PROCESS_CTRL | 0x1018 | BPA bypass | 0x0280 | 347 |

## How BPA Works

The “bad” pixels are corrected by comparing their value with the value of other pixels in the local vicinity. Data from the same color pixels to the left and right of the pixel under test, along with data from the other color pixels are used. Gradients are determined to see if the pixel under test is a local minimum or maximum. If not, the pixel is good.

If it is a local minimum or maximum, the gradient in the vertical and the two diagonal directions is checked to see if it is also a local minimum or maximum in those directions. If not, then the pixel is probably on a thin line and will be left uncorrected.

If it is still a local minimum or maximum, then the pixel value is compared to the predicted value, using the local gradients. This comparison is done using three values: OUTLIER and PEDESTAL, (stored in the BPA_OUTL_PED register) and SCALE_FACTOR, which are stored in the BPA_SF_GTHRESH register.

The pixel is bad if:

- Value is greater than the predicted value plus the outlier value
- Value is greater than the predicted value times the scale factor plus the pedestal value
- Value is less than the predicted value minus the outlier value
- Value is less than the predicted value times the inverse of the scale factor minus the pedestal

If the pixel is determined to be “bad”, then the pixel value is replaced with the median of the local neighbors of the same color.

## Start of Frame / End of Frame Codes

The ADCM-2700 camera module has the ability to add a data byte before and after the data frame.

**Start of frame/end of frame registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | SOF/EOF enable, bits 13 and 14 | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | SOF/EOF enable, bits 13 and 14 | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | SOF/EOF enable, bits 13 and 14 | 0x9019 | 338 |
| SOF_CODES | 0x0146 | Start of frame codes, video and still modes | 0xfeff | 223 |
| EOF_CODES | 0x0148 | End of frame codes, good and bad frame | 0x0100 | 224 |
| SOF_CODES_W | 0x100c | Start of frame codes, current mode | 0x00ff | 341 |
| EOF_CODES_W | 0x100e | End of frame codes, good and bad frame | 0x0100 | 342 |

## Expert Pixel Timing Registers

The expert pixel timing registers are use to fine tune the pixel reset and exposure timing. Do not write to these registers directly.

**NOTE** It is possible that by modifying these registers, the sensor will be programmed into a non-functional state. If this occurs reset the camera module.

The ADCM-2700 sensor timing controller has flexible features to determine the pulse timing for the pixel array. The exposure period (exposure reset) and sample period are both dependent on these register settings.

Three active control signals are generated, which are reset, preset and ground referenced reset. The internal timing controller determines when each sequence begins.

In general, the exposure sequence consists of two edges (one pulse) for each signal, while the sample sequence consists of three edges (one pulse plus a post pulse delay). Each sequence continues until the last signal has completed the required edges.

**Exposure sequence registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| EXP_RST_E0 | 0x08f1 | Exposure, reset edge 0 | 0x03 | 317 |
| EXP_RST_E1 | 0x08f0 | Exposure, reset edge 1 | 0x10 | 316 |
| EXP_PRST_E0 | 0x08fb | Exposure, preset edge 0 | 0x06 | 326 |
| EXP_PRST_E1 | 0x08fa | Exposure, preset edge 1 | 0x02 | 325 |
| EXP_GR_E0 | 0x08de | Exposure, ground reference edge 0 | 0x06 | 308 |
| EXP_GR_E1 | 0x08dc | Exposure, ground reference edge 1 | 0x10 | 307 |

**Sample sequence registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SMP_RST_E0 | 0x08ed | Sample, reset edge 0 | 0x07 | 314 |
| SMP_RST_E1 | 0x08ec | Sample, reset edge 1 | 0x10 | 313 |
| SMP_RST_E2 | 0x08eb | Sample, reset edge 2 | 0x04 | 312 |
| SMP_PRST_E0 | 0x08f7 | Sample, preset edge 0 | 0x0a | 323 |
| SMP_PRST_E1 | 0x08f6 | Sample, preset edge 1 | 0x02 | 322 |
| SMP_PRST_E2 | 0x08f5 | Sample, preset edge 2 | 0x00 | 321 |
| SMP_GR_E0 | 0x08d9 | Sample, ground reference edge 0 | 0x0a | 305 |
| SMP_GR_E1 | 0x08d8 | Sample, ground reference edge 1 | 0x10 | 304 |
| SMP_GR_E2 | 0x08d7 | Sample, ground reference edge 2 | 0x00 | 303 |

**Pixel control polarity registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| GR_POL | 0x08df | Ground reference polarity | 0xd3 | 310 |
| PRESET_POL | 0x08fd | Preset polarity | 0xd3 | 328 |
| RESET_POL | 0x08f3 | Reset polarity | 0xd3 | 319 |

# Test Pattern Generator

The ADCM-2700 camera module has the ability to generate test patterns.

**Test pattern register**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| DATA_GEN | 0x105e | Test pattern generator | 0x0000 | 365 |

There are six different test patterns. The data rate is adjustable. Four of the patterns have the option of motion between frames.

## Normal Mode

In normal mode, no test pattern is generated and the image from the sensor is sent to the image pipeline.

## White Mode

In white mode, all of the pixels are set to 255, 255, 255. Motion is a slow fade to black, with a jump back to white.



**Figure 13** White mode test pattern

## Random Mode

In random mode, all of the pixels are set by a pseudo random number generator. Motion is a new random number sequence for each frame.

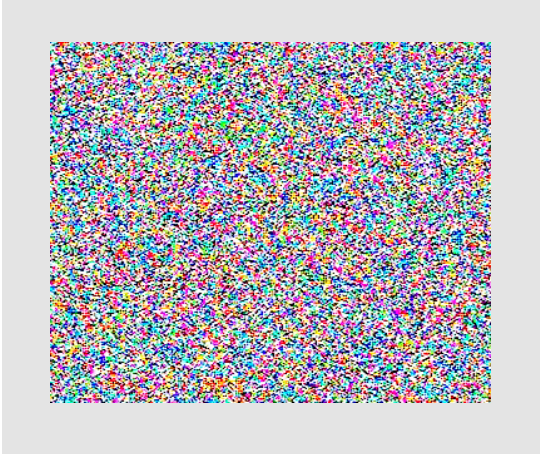


**Figure 14** Random mode test pattern

## Sum of Coordinates Mode

In sum of coordinates mode, the sum of the pixel coordinates is output as the brightness value. Motion is an increasing offset to the first pixel, which results in horizontal motion of the diagonal lines.

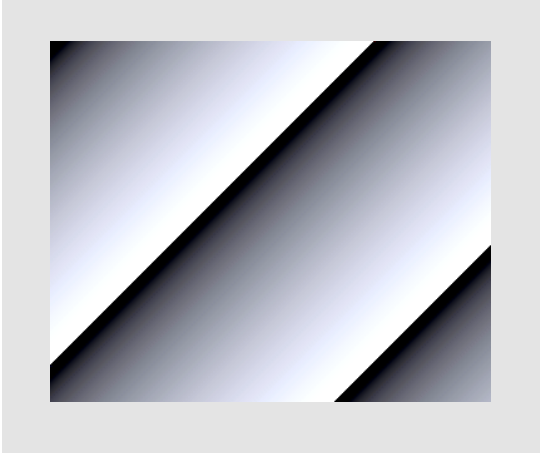


**Figure 15** Sum of coordinates mode test pattern

## Eight Pixel Wide Border Mode

In eight pixel wide border mode, the eight border pixels (top, bottom, left and right) are white, with all other pixels black. Motion increases the number of border pixels that are white. Once all of the pixels are white, the image stays white for a time and then jumps back to an eight pixel border.

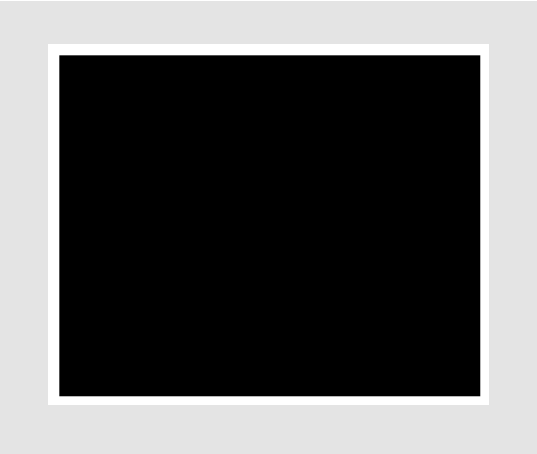


**Figure 16** Eight pixel border mode test pattern

## Checkerboard Mode

In checkboard mode, a variable checkerboard is displayed and no motion is possible.

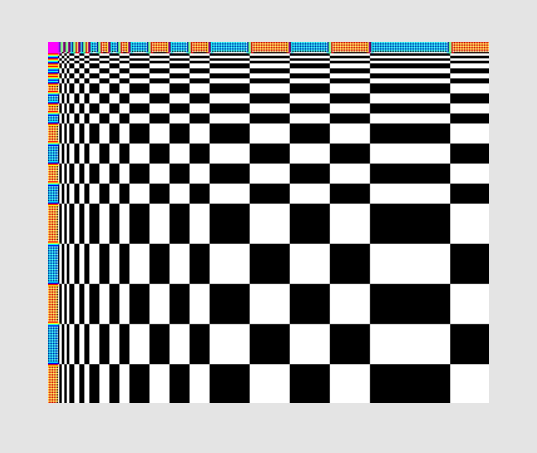


**Figure 17** Checkerboard mode test pattern

### Color Bars Mode

In color bars mode, the color bars are displayed and no motion is possible.

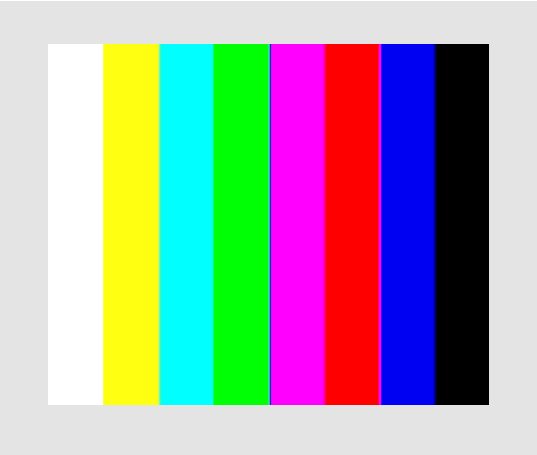


**Figure 18** Color bars mode test pattern

## Row Processing Time

The ADCM-2700 camera module has registers that report the current row processing time, given in sensor clocks. The values in these registers are calculated by the camera controller using the current settings and should be considered read only.

**Row processing time registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| RPT_V | 0x0114 | Row processing time, video mode | 0x0546 | 196 |
| RPT_S | 0x0134 | Row processing time, still mode | 0x0546 | 214 |

## Horizontal and Vertical Blanking Intervals

Horizontal and vertical blanking times can be added to the output data to fine tune the exposure and frame rate. These registers are controlled by the firmware and under normal operation, should not be changed.

**Horizontal and vertical blanking registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| HBLANK_V | 0x010a | Sensor HBLANK value, video mode | 0x0000 | 189 |
| HBLANK_S | 0x012a | Sensor HBLANK value, still mode | 0x0000 | 207 |
| HBLANK | 0x0819 | HBLANK value | 0x00 | 292 |
| VBLANK_V | 0x010c | Sensor VBLANK value, video mode | 0x0000 | 190 |
| VBLANK_S | 0x012c | Sensor VBLANK value, still mode | 0x0000 | 208 |
| VBLANK | 0x081a | VBLANK value | 0x00 | 293 |

HBLANK is the number of sensor clock cycles that are added to the row processing time, after the row sample period and before the A/D converter latency. See “Basic Timing Controller Operations" on page 407 for additional information.

VBLANK is the number of row processing periods that are added to increase the vertical blanking period independently of exposure time. See “First-Frame and Inter-Frame Delay Periods" on page 404 for additional information.

# 2 External Clocks / Using the PLL



## Overview

On power-up, the camera module's internal clocks are derived from the MCLK input, using divisors to divide down MCLK to the serial control, sensor and image pipeline clock frequencies. The default values of the divisors assume that the MCLK input frequency is 13 MHz. MCLK can vary from 8 to 26 MHz.

The ADCM-2700 camera module also includes a programmable phase lock loop (PLL) to allow the camera module to use a different frequency clock than the MCLK provided to the module. The PLL default settings are also for a 13 MHz input clock.

### Clock Ratios – Image Pipeline and Sensor

Since the image sensor and image pipeline must communicate with each other, there are limits on the differences in their respective clock rates. The image sensor portion of the IC will output one pixel for every clock cycle, the image pipeline needs two clock cycles to read and process the pixel.

# Clocks

The ADCM-2700 camera module uses an external clock (MCLK), with divisor registers to determine the final image pipeline and sensor clocks. These registers are normally controlled by the simple control registers.

**Clock registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CLK_PER | 0x0006 | MCLK input frequency | 0x32c8 | 144 |
| FRAME_RATE | 0x0012 | Output frame rate | 0x0096 | 152 |
| I_CLK_DIV | 0x0080 | Initial clock divisor:<br>PRESCALE = I_CLK_DIV[2:0] + 1 = 2 | 0x0001 | 166 |
| CTL_CLK_DIV | 0x0082 | Clock divisors for control and serial interface:<br>CTL_CLK = CTL_CLK_DIV[2:0] + 1 = 1 | 0x4000 | 167 |
| SEN_CLK_DIV | 0x0084 | Sensor clock divisors:<br>SEN_CLK_DIV[2:0] = CLK_DIV_V[2:0]<br>or<br>SEN_CLK_DIV[2:0] = CLK_DIV_S[2:0] | 0x0000 | 168 |
| IP_CLK_DIV | 0x0086 | Image pipeline clock divisors:<br>IP_CLK_DIV[2:0] = CLK_DIV_V[10:8]<br>or<br>IP_CLK_DIV[2:0] = CLK_DIV_S[10:8] | 0x0000 | 169 |
| PLL_CTRL | 0x0090 | PLL control register | 0x0024 | 174 |
| PLL_DIV_L | 0x0092 | PLL large divisors:<br>VCO_DIV_L = PLL_DIV_L [15:8] = 0x1d = 29<br>REF_DIV_L = PLL_DIV_L [15:8] = 0x09 = 9 | 0x1d09 | 175 |
| PLL_DIV_S | 0x0094 | PLL small divisors:<br>VCO_DIV_S = PLL_DIV_S [15:8] = 0x03 = 3<br>REF_DIV_S = PLL_DIV_S [15:8] = 0x00 = 0 | 0x0200 | 176 |
| CPP_V | 0x0108 | Number of sensor clocks per pixel – video | 0x0002 | 188 |
| CLK_DIV_V | 0x0118 | Clock divisors, video mode:<br>SEN_CLK_DIV[2:0] = CLK_DIV_V[2:0]<br>IP_CLK_DIV[2:0] = CLK_DIV_V[10:8] | 0x0000 | 198 |
| PARALLEL_CTRL_V | 0x011a | VCLK output divisor – video | 0x0003 | 199 |
| CPP_S | 0x0128 | Number of sensor clocks per pixel – still | 0x0002 | 206 |
| CLK_DIV_S | 0x0138 | Clock divisors, still mode:<br>SEN_CLK_DIV[2:0] = CLK_DIV_S[2:0]<br>IP_CLK_DIV[2:0] = CLK_DIV_S[10:8] | 0x0000 | 216 |
| PARALLEL_CTRL_S | 0x013a | VCLK output divisor – still | 0x0000 | 217 |

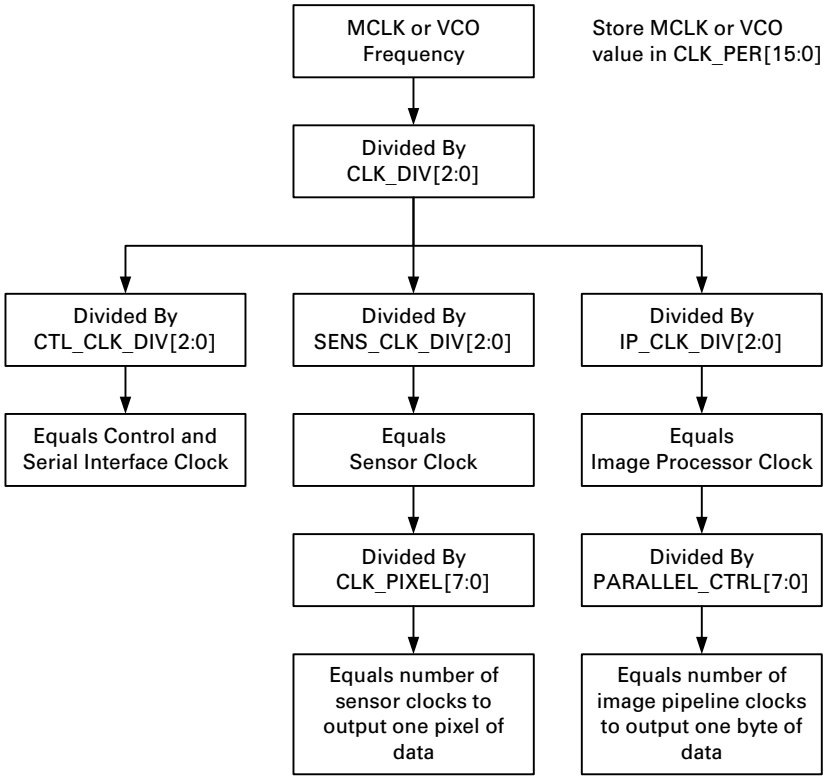
**Clock registers (continued)**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CLK_PIXEL | 0x080e | Number of sensor clocks per pixel – working register:<br>CLK_PIXEL[7:0] = CPP_V[7:0]<br>or<br>CLK_PIXEL[7:0] = CPP_S[7:0] | 0x02 | 282 |
| PARALLEL_CTRL | 0x100a | VCLK output divisor – working register:<br>PARALLEL_CTRL[7:0] = PARALLEL_CTRL_V[7:0]<br>or<br>PARALLEL_CTRL[7:0] = PARALLEL_CTRL_S[7:0] | 0x0000 | 340 |

The simple control registers use the CLK_PER, FRAME_RATE and MAX_SCLK registers to determine the slowest possible clocks that will result in correct operation for the currently selected window size and output format. The slowest possible clocks result in the lowest power consumption.

The PLL is not controlled by the simple control registers. To use the PLL, program the PLL registers and set the new VCO frequency into CLK_PER so that the simple control registers will correctly determine the divisors.

## Clock Divisors




| LOADED TO WORKING REGISTERS WHEN "SNAP" BIT IS SET | LOADED TO WORKING REGISTERS WHEN "RUN" BIT IS SET | WORKING REGISTERS |
|---|---|---|
| CPP_S[7:0]<br>CLK_DIV_S[2:0]<br>CLK_DIV_S[10:8]<br>PARALLEL_CTRL_S[7:0] | CPP_V[7:0]<br>CLK_DIV_V[2:0]<br>CLK_DIV_V[10:8]<br>PARALLEL_CTRL_V[7:0] | CLK_PIXEL[7:0]<br>SENS_CLK_DIV[2:0]<br>IP_CLK_DIV[2:0]<br>PARALLEL_CTRL[7:0] |

| REGISTER | VALID DIVISORS |
|---|---|
| CLK_DIV | 1, 2, 3, 4, 5, 6, 7, 8 |
| CTL_CLK_DIV | 1, 2, 4, 6, 8, 10, 12, 14 |
| SENS_CLK_DIV | 1, 2, 4, 6, 8, 10, 12, 14 |
| IP_CLK_DIV | 1, 2, 4, 6, 8, 10, 12, 14 |
| CLK_PIXEL | 1 to 255 |
| PARALLEL_CTRL | 1 to 255 |

**Figure 19** ADCM-2700 clock divisors

## Enabling and Disabling the PLL

The PLL allows clock frequencies to be used that are not available in external clock mode which is the default for the ADCM-2700.

To enable the PLL, calculate the PLL coefficients, load them into the PLL registers. Load the VCO frequency into the CLK_PER register. Enable the PLL by setting the PLL_EN bit in the PLL_CTRL register.

To disable the PLL, (clear the PLL_EN bit in PLL_CTRL). Load the MCLK value into the CLK_PER register. The camera module is now in external clock mode.

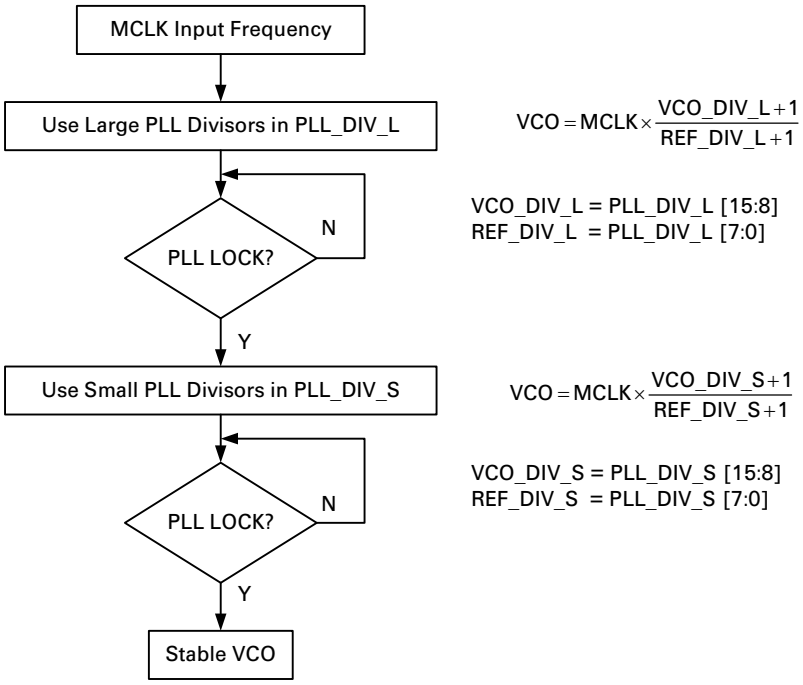



**Figure 20** ADCM-2700 PLL

## PLL Theory of Operation

The PLL VCO frequency obeys the following equation:

$$\frac{VCO}{VCO\_DIV + 1} = \frac{MCLK}{REF\_DIV + 1}$$

For a given MCLK, VCO equals:

$$VCO = MCLK \times \frac{VCO\_DIV + 1}{REF\_DIV + 1}$$

The range of valid VCO values is 1 to 135 MHz.

### VCO_DIV and REF_DIV Values

There are two sets of VCO_DIV and REF_DIV values stored in tow sets of registers, PLL_DIV_L and PLL_DIV_S, large and small values. The larger divisors have a smaller probability of aliasing the VCO frequency, but will have higher jitter. The smaller divisors can result in the VCO aliasing to the wrong frequency, but the frequency will jitter less.

When the PLL is enabled (bit PLL_EN in PLL_CTRL), the PLL starts by using the larger divisor values to lock in to the desired frequency. Once the frequency is locked, the PLL switches to the smaller divisors to reduce the amount of jitter. Since the PLL is locked at the desired frequency, the smaller divisors do not cause aliasing.

### Default VCO_DIV and REF_DIV Values

The following table shows the VCO_DIV and REF_DIV default values.

**VCO_DIV and REF_DIV values**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| PLL_DIV_L | 0x0092 | PLL large divisors:<br>VCO_DIV_L = PLL_DIV_L [15:8] = 0x1d = 29<br>REF_DIV_L = PLL_DIV_L [15:8] = 0x09 = 9 | 0x1d09 | 175 |
| PLL_DIV_S | 0x0094 | PLL small divisors:<br>VCO_DIV_S = PLL_DIV_S [15:8] = 0x03 = 3<br>REF_DIV_S = PLL_DIV_S [15:8] = 0x00 = 0 | 0x0200 | 176 |

This results in the MCLK multipliers:

$$\text{Large Values} \quad \frac{VCO\_DIV + 1}{REF\_DIV + 1} = \frac{30}{10} = 3$$

$$\text{Small Values} \quad \frac{VCO\_DIV + 1}{REF\_DIV + 1} = \frac{3}{1} = 3$$

Care should be taken to have the result for the large and small values be close to each other, otherwise the frequency will shift when the PLL changes from the larger to smaller divisors.

## Determining Clock Divisors

Once the MCLK or the VCO clock frequency is known, (for VCO see “PLL Theory of Operation" on page 64) the clock divisors can be determined. The ADCM-2700 has three sets of clock divisors: video mode, still mode and current working mode. Only write to the video and still mode registers. Depending on which mode the camera is in, the ADCM-2700 copies the relevant registers to the current working mode registers.

For a chart showing the relationship of the divisors to the input clock (MCLK or VCO), see Figure 19 on page 62.

### Initial Divisor

Only the initial divisor CLK_DIV allows for division by odd values. Valid values are from 1 to 8. The output from CLK_DIV is passed to the control and serial interface clock, the sensor clock and the image pipeline clock.

### Control and Serial Interface Clocks

Bits [2:0] in CTL_CLK_DIV (address 0x0082, see page 167) determine the divisor for the control and synchronous serial interface clocks. The value used is 2 times the bit value, n. If n = 0, the divisor is 1. Valid values are 1, 2, 4, 6, 8, 10, 12 and 14.

### Sensor Clock – Main

The value of the sensor clock divisor register, SENS_CLK_DIV (address 0x0084, see page 168) is controlled by either CLK_DIV_V (bits [2:0], address 0x0118, see page 198) or CLK_DIV_S (bits [2:0], address 0x0138, see page 216) depending on the current mode, video or still.

The value used is 2 times the bit value, n. If n = 0, the divisor is 1. Valid values are 1, 2, 4, 6, 8, 10, 12 and 14.

### Sensor Clock, Clocks per Pixel

The number of sensor clocks needed to output one pixel of data is controlled by CLK_PIXEL (8-bit sensor register, address 0x080e, see page 282). The CLK_PIXEL value is controlled by either CPP_V (address 0x0108, see page 188) or CPP_S (address 0x0128, see page 206) depending on the current mode. Valid values are 1 to 255.

### Image Pipeline Clock, Main

The image pipeline clock divisor register IP_CLK_DIV (address 0x0086, see page 169) is controlled by either CLK_DIV_V (bits [10:8], address 0x0118, see page 198) or CLK_DIV_S (bits [10:8], address 0x0138, see page 216) depending on the current mode, video or still.

The value used is 2 times the bit value, n. If n = 0, the divisor is 1. Valid values are 1, 2, 4, 6, 8, 10, 12 and 14.

### Image Pipeline Clock, Output Divisor

The data output clock VCLK, is derived from the image pipeline clock, divided by bits [7:0] in the PARALLEL_CTRL register (address 0x100a, see page 340). The value of PARALLEL_CTRL is controlled by PARALLEL_CTRL_V (address 0x011a, see page 199) or by PARALLEL_CTRL_S (address 0x013a, see page 217).

The value used is 1+ the value in the register. Valid register values are 0 to 255.

## Clock Dithering

The VCO can be dithered to improve EMI performance. Set bit PLL_DE in PLL_CTRL.

# 3 Serial Control Interface



The ADCM-2700 camera module uses a two-wire serial control interface for non-image data transmission. Image frames are transmitted through a separate parallel interface.

## Master and Slave Operation

The control interface requires one device to act as a master, with all other devices acting as slaves (the ADCM-2700 camera module can only operate as a slave). The ADCM-2700 camera module has a device address of 0x53.

## Serial Transfer Clock and Serial Data Signals

The serial control interface uses two signals: a serial transfer clock (SCLK) and a serial data (SDATA) signal. Always driven by the master, SCLK synchronizes the serial transmission of data bits on SDATA. The frequency of SCLK may vary throughout a transfer as long as the timing is greater than the minimum timing.

Normally, the host drives SDATA but the module drives SDATA only under two conditions. First, when responding with an acknowledge bit after receiving data from the host, or second, when sending data to the host at the host's request. Data is sent in 8-bit packets.

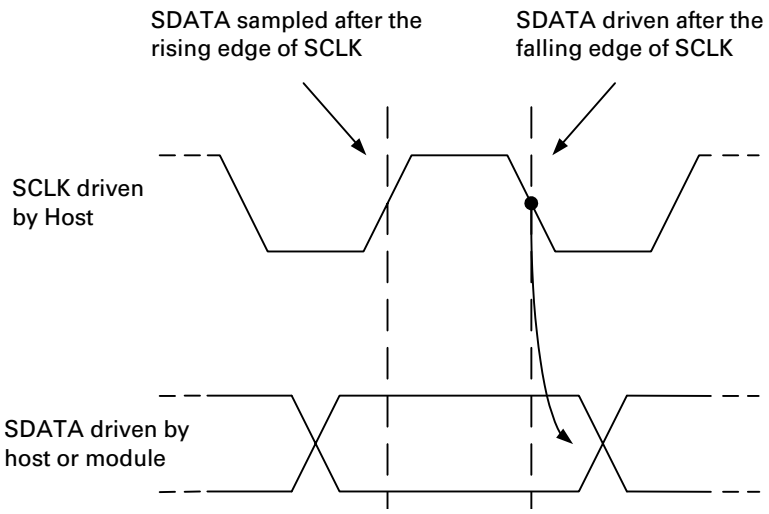



**Figure 21** Synchronization of two-wire serial interface signals

# SCLK and SDATA Timing

This section explains SCLK and SDATA timing.

## Module Driven SDATA

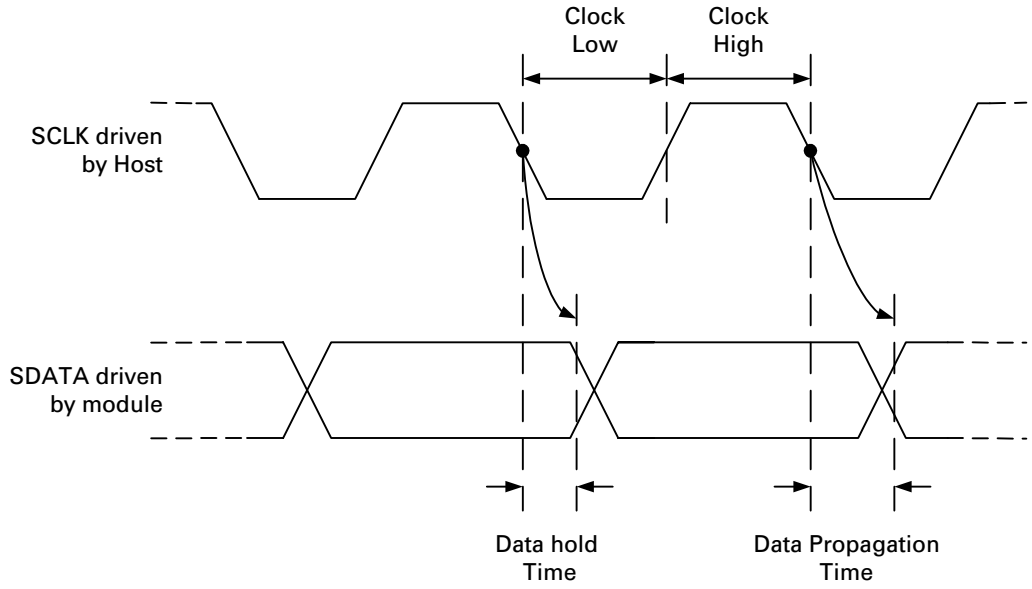



**Figure 22** Serial interface timing definitions for module driven SDATA

**Serial interface timing specifications for module driven SDATA**

| Condition | Internal Clock Cycles | Description |
|---|---|---|
| Minimum data hold time | 1 | Cycles that SDATA is held stable after SCLK falls |
| Maximum data propagation time | 2 + rise and fall time | Cycles after SCLK falls before SDATA becomes stable. (Rise time is controlled by external devices and should be >> fall time.) |
| SCLK low | 4 | Clock at logic level 0 |
| SCLK high | 4 | Clock at logic level 1 |

### Host Driven SDATA

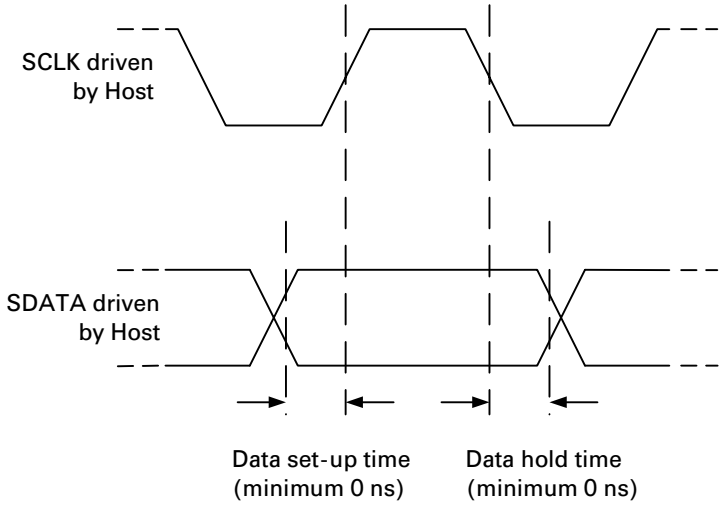



**Figure 23** Serial interface timing definitions for host driven SDATA

## Start and Stop of Synchronous Operation

The host initiates and terminates all data transfers. Data transfers are initiated by driving SDATA from high to low while holding SCLK high. Data transfers are terminated by driving SDATA from low to high while SCLK is held high.

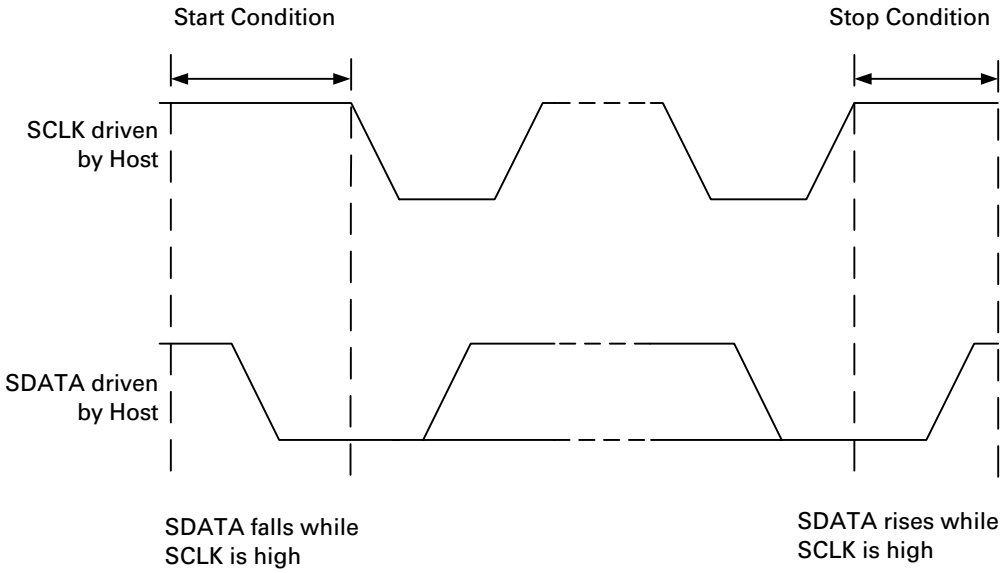



**Figure 24** Two-wire serial interface start and stop conditions

## Acknowledge / Not Acknowledge Bit

After a start condition, a single acknowledge/not acknowledge bit follows each 8-bit data packet. The device receiving the data drives the acknowledge/not acknowledge signal on SDATA. Acknowledge (ACK) is defined as “0” and not acknowledge (NAK) is defined as “1”.

## Host Driven Packets

The host initiates all data transmission with a start condition. If there is a device address match, the module then responds to each 8-bit data transmission with an acknowledge signal (SDATA = 0). Data is transmitted with the most significant bit first.

To terminate the transfer of host driven packets, the host follows the module's ACK with a stop condition. The host can also issue a start condition after the module's ACK if it wants to start a new data transfer.

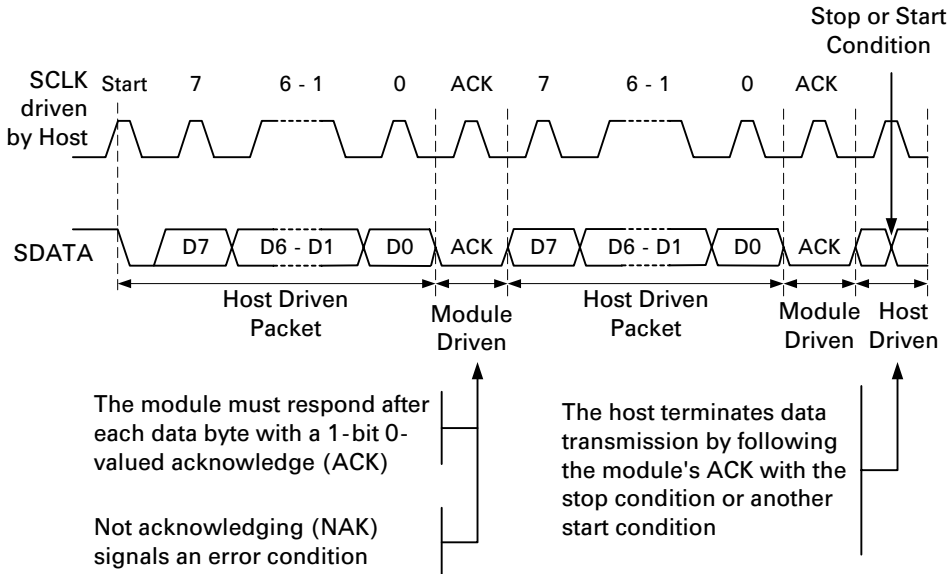



**Figure 25** Two-wire serial interface – host driven packets

## Module Driven Packets

By request of a host, the module acknowledges a read request and then outputs a data byte transmitting the most significant bit (7) first. If the host wants to continue the data transfer, the host acknowledges the module. If the host wants to terminate the transfer, it responds with not acknowledge (SDATA = 1) and then drives SDATA to generate a stop condition. The host can also drive a start condition if it wants to begin a new data transfer with the same module.

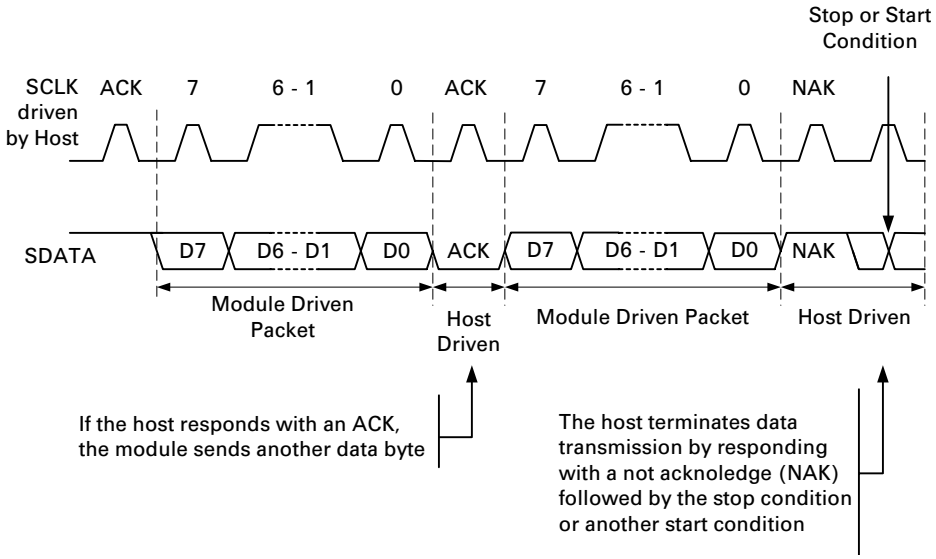



**Figure 26** Two-wire serial interface – module driven packets

# Packet Formats

Read and write operations between the host and a module use three types of host driven packets and one type of module driven packet. All packets are 8 bits long with the most significant bit first and followed by an acknowledge bit.

## Command Packets (MC)

Contain a 7-bit module device address and an active low read/write bit (R/$\overline{W}$). The device address of the ADCM-2700 camera module is 0x53.

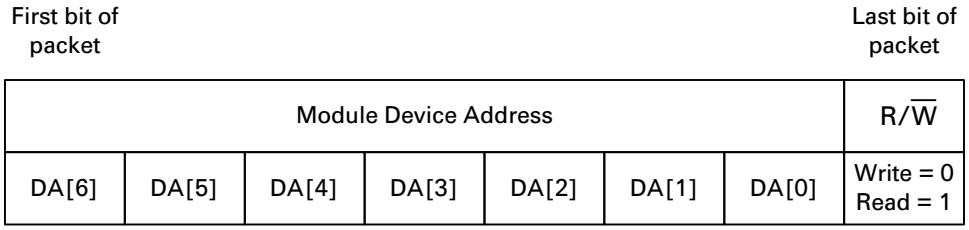



**Figure 27** Command packet (MC)

## Address Packets (MA)

Contain a 7-bit module address or a block switch code and an active low read/write bit (R/$\overline{W}$).

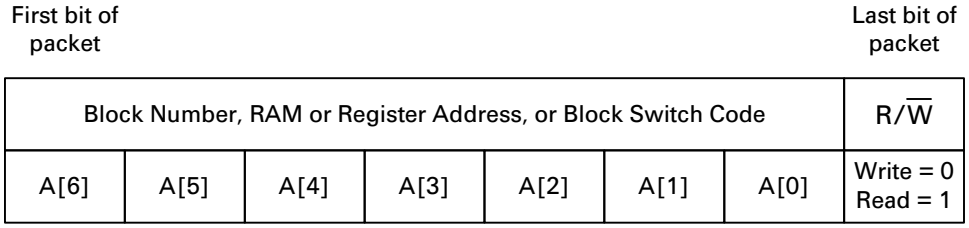



**Figure 28** Address packet (MA)

As discussed in Chapter 1, “Addressing RAM and Registers”, RAM and register space is divided into blocks of 64 addressable locations. All addresses in a block have a least significant bit (A[0]) of 0. Writing the base address of a block to the ADCM-2700 camera module requires a special 2-packet sequence.

First, write an address packet with A[6:0] = 0x7f (the block switch code) so that A[0] = 1. This signals that the following packet, a data packet, contains a new block address. For further information about setting the block number for read / write operations, see Chapter 3, “16-Bit Read and Write Examples”.

### Data Packet (MD or SD)

Contains 8 data bits and may be sent by the host or the camera module.

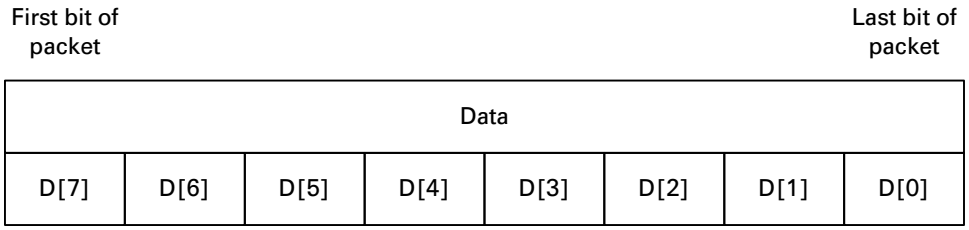



**Figure 29** Data packet (MD or SD)

**NOTE** Although the data packets are eight bits long, in general, all data sent to and received from the ADCM-2700 camera module is in 16-bit words. Two data packets are needed for ONE data transfer. The only exception to this is for the sensor addresses 0x0800 through 0x8fd. Data to and from these addresses are 8-bit.

## 16-Bit Read and Write Examples

The following sequence of examples shows how to write and read data to and from the ADCM-2700 camera module. The host initiates and terminates all communication and the host sends a MC packet after driving the start condition. The ADCM-2700 camera module will respond to the host if the MC packet contains the module device address (0x053).

In the following examples, the camera module is shown always responding with an ACK to transmissions from the host. If at any time a NAK is received, the host should transmit a stop bit and then retransmit the data to the camera module, starting with the start bit and the device address.

## Example 1: Specify the ADCM-2700 Module Block Number

The host controls the block number that is used in all RAM and register addressing. As mentioned in Chapter 3, “Packet Formats”, writing a block number to the ADCM-2700 camera module requires a special two-packet sequence.

The first packet, a master address packet, contains the block switch code (bits 7:1 = 0x7f) and the second packet, a master data packet, contains the new block number. The following example sets the block number to 0x02.

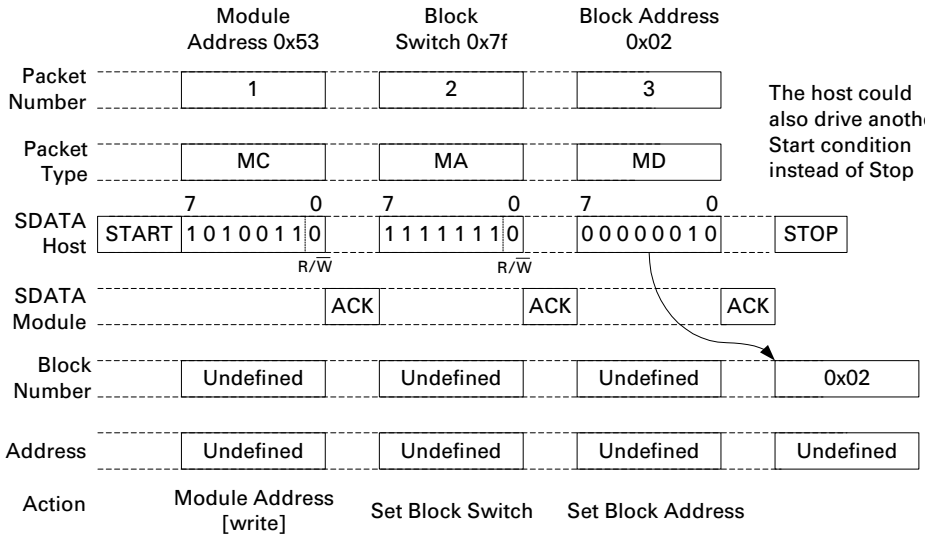



**Figure 30** Setting the module block number to 0x02

| Serial Control Data | Hex |
|---|---|
| Sent | a6, fe, 02 |
| Received | – |

## Example 2: Writing Data to the ADCM-2700

**Addressing:** Once the block number has been set, subsequent read and write operations need only specify the starting address offset. The ADCM-2700 camera module automatically increments its internal address pointer after reading or writing a byte (data packet). However, the auto-increment function cannot change the block number when it reaches the maximum address in a block and needs to start with the first address of the next block. The block number must be changed as indicated in example 1.

**Data width, data order:** All data either read or written to the ADCM-2700 camera module is 16 bits wide (two data packets). The host devices should always write or read an even numbers of bytes. Within the 16-bit data word, the least significant byte is first. Within each byte (data packet), the most significant bit is first.

**Writing to OUTPUT_CTRL_V and PROC_CTRL_V:** In the following example, the host writes the default values to the registers: 0x9019 to the OUTPUT_CTRL_V and 0x0680 to the PROC_CTRL_V. These registers occupy consecutive register locations (block 2, address offsets 0x10 and 0x12) so the host only needs to write the starting address offset (0x010).

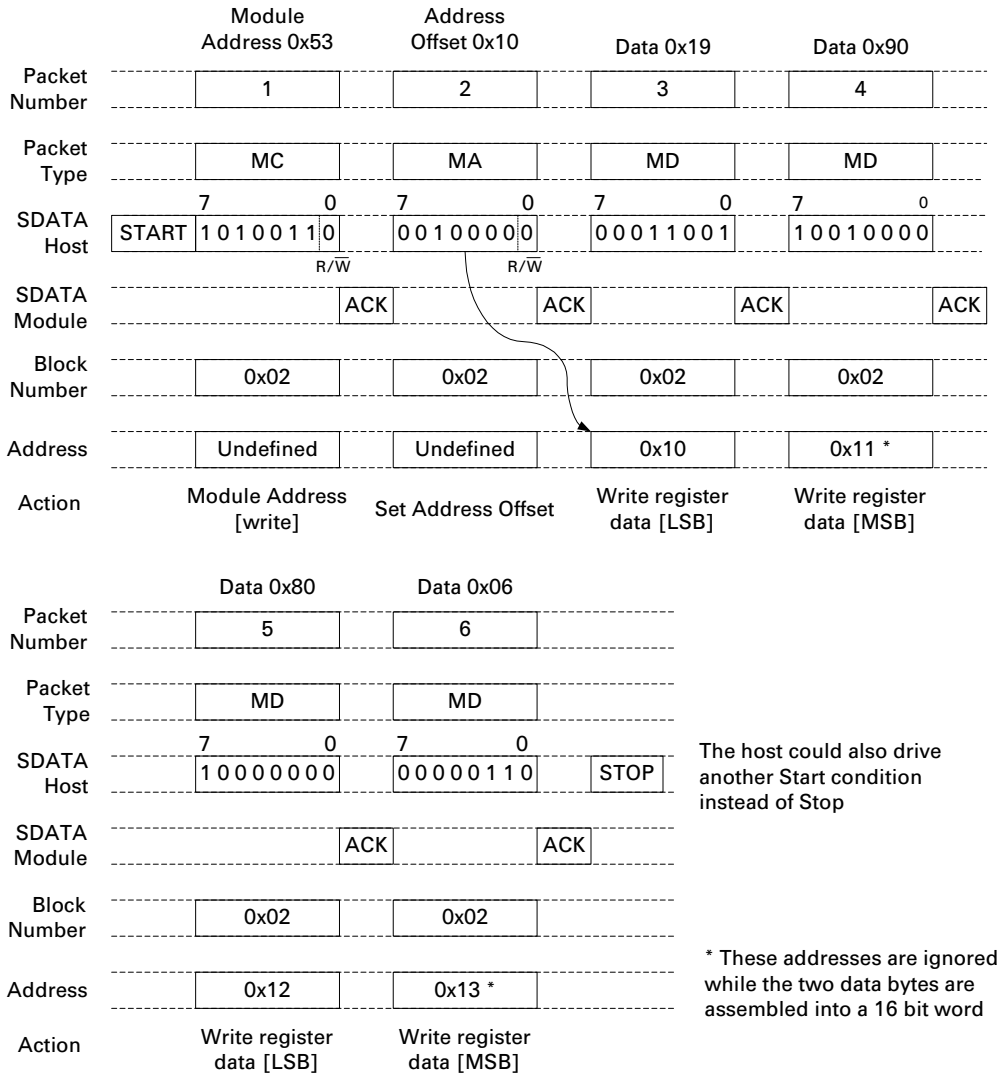



**Figure 31** Four byte write to block 0x02, address offset 0x10: OUTPUT_CTRL_V and PROC_CTRL_V registers

| Serial Control Data | Hex |
|---|---|
| Sent | a6, 20, 19, 90, 80, 06 |
| Received | – |

## Example 3: Reading Data from the ADCM-2700

Similar to the write operation, a read operation will need to specify the block number if the desired memory location or register resides in block different from what was last written to the ADCM-2700 camera module.

To set the block number prior to reading data, follow example 1 (see page 75), and substitute the proper block number in packet number 3 (the master data packet). Once the block number has been set or if the current block number is correct, the host writes the starting address for the read operation.

Just like the write operation, the ADCM-2700 camera module will auto-increment its address pointer when the host is reading data. Once the address has been written to the module, the host sends a control packet requesting to read data. The ADCM-2700 camera module responds by sending data packets until the host responds with a not acknowledge (NACK) signal.

**Reading from CSC_00_V:** In the following example, the CSC_00_V register is in block 3, address offset 0x10 and the default value is 0x0026. Since the previous example used a block number of 2, the host first writes a new block address (3).

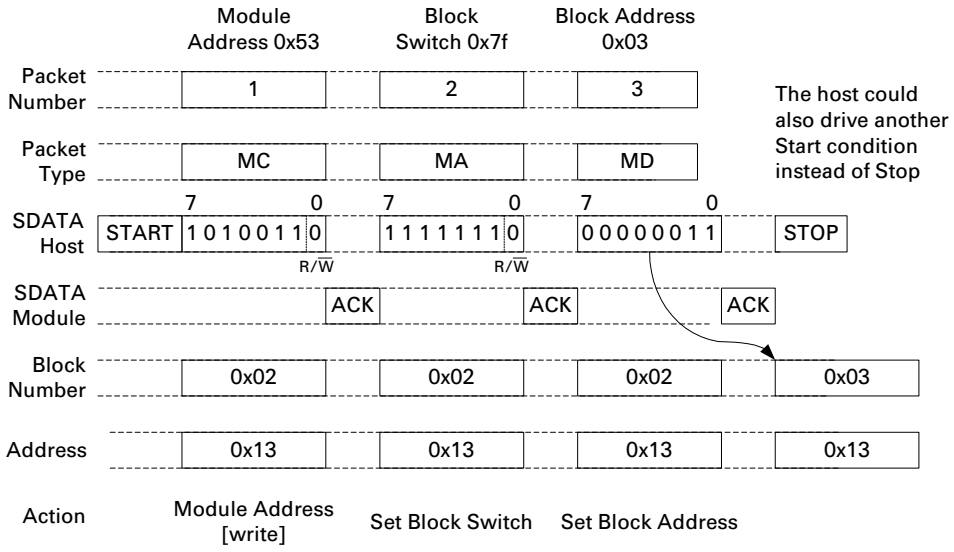



**Figure 32** Changing the block number to 0x03

| Serial Control Data | Hex |
|---|---|
| Sent | a6, fe, 03 |
| Received | – |

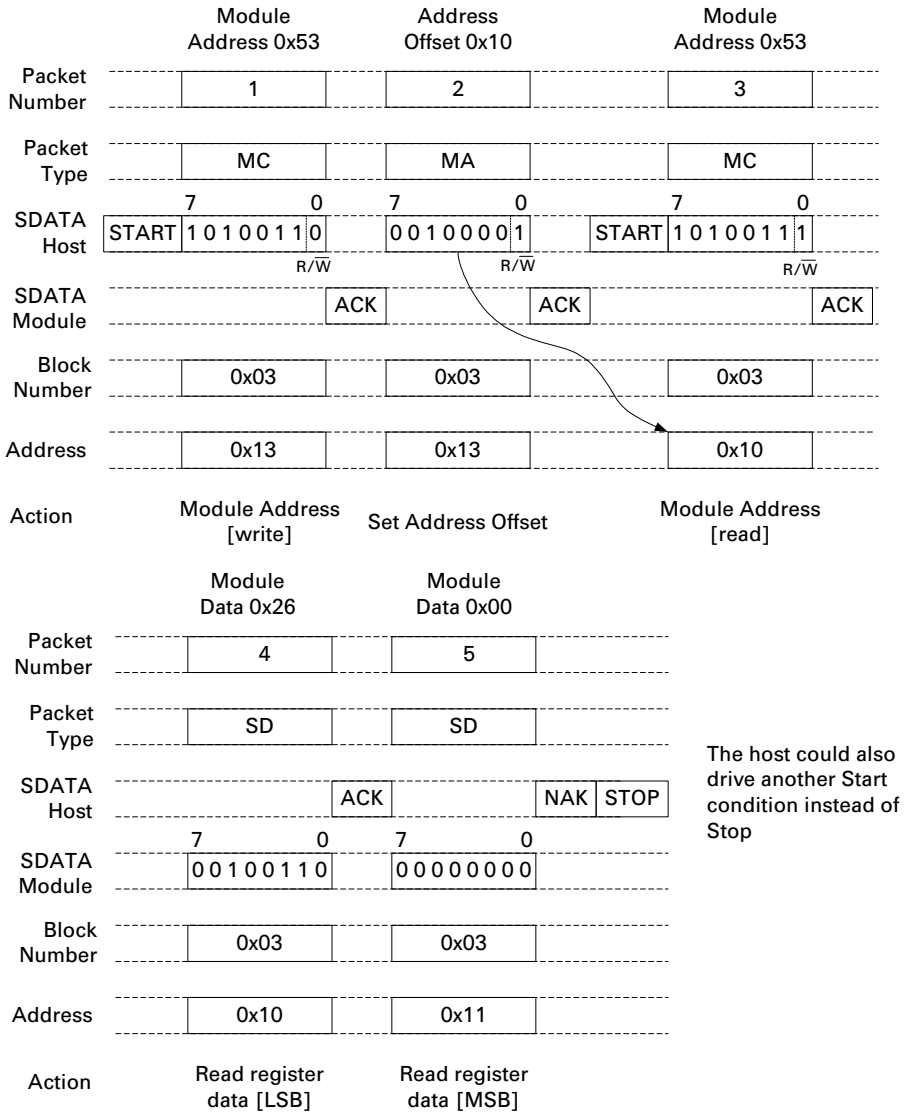



**Figure 33** Reading two bytes from the module, block 0x03, address 0x10: CSC_00_V register

| Serial Control Data | Hex |
|---|---|
| Sent | a6, 21, a7 |
| Received | 26, 00 |

# 8-Bit Read and Write Examples

## Sensor Registers

The following sequence of examples show how to read and write data to the 8-bit sensor registers. The host initiates and terminates all communication and also sends a MC packet after driving the start condition. The ADCM-2700 camera module will respond to the host if the MC packet contains the ADCM-2700 camera module device address (0x053).

The module is shown always responding with an ACK to transmissions from the host in the following examples. If at any time a NAK is received, the host should transmit a stop bit and then retransmit the data to the module starting with the start bit and the device address.

## Example 4: Specify the Module Block Number

The block number for the sensor addresses is either 0x10 or 0x11. The block number will be set to 0x10 for the following examples.

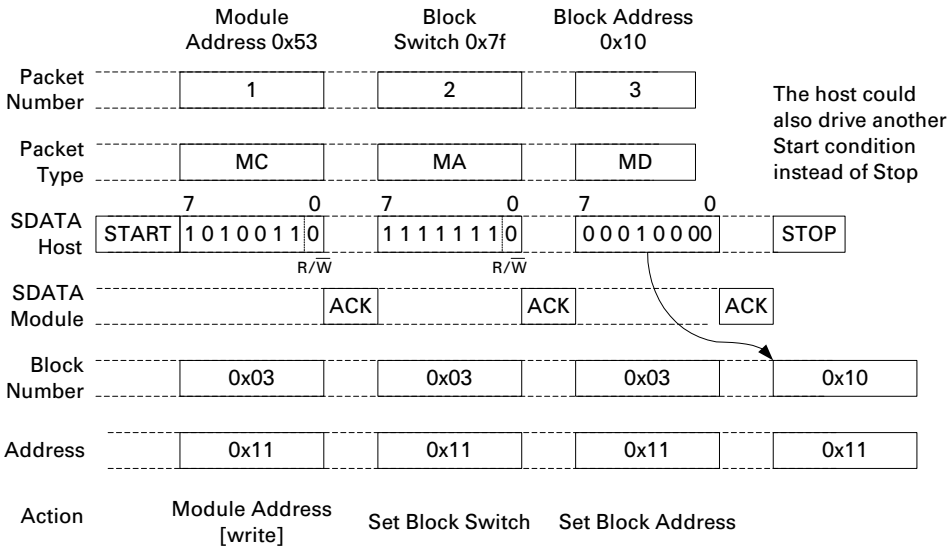



**Figure 34** Setting the block address to 0x10

| Serial Control Data | Hex |
|---|---|
| Sent | a6, fe, 10 |
| Received | – |

## Example 5: Writing Data to the ADCM-2700 Sensor Registers

**Addressing:** Once the block number has been set, subsequent read and write operations need only specify the starting address offset. The ADCM-2700 camera module automatically increments its internal address pointer after reading or writing a byte (data packet). However, the auto-increment function cannot change the block number when it reaches the maximum address in a block and needs to start with the first address of the next block. The block number must be changed as indicated in example 4.

**Data width, data order:** All data either read or written to the ADCM-2700 sensor registers eight bits wide (one data packets). Within each byte (data packet), the most significant bit is first.

**Writing to FWCOL:** In this example, the host writes a value of 0x06 to the FWCOL register, address 0x080a. This register is in block 0x10, so if example 4 was done, the block number is already set correctly.

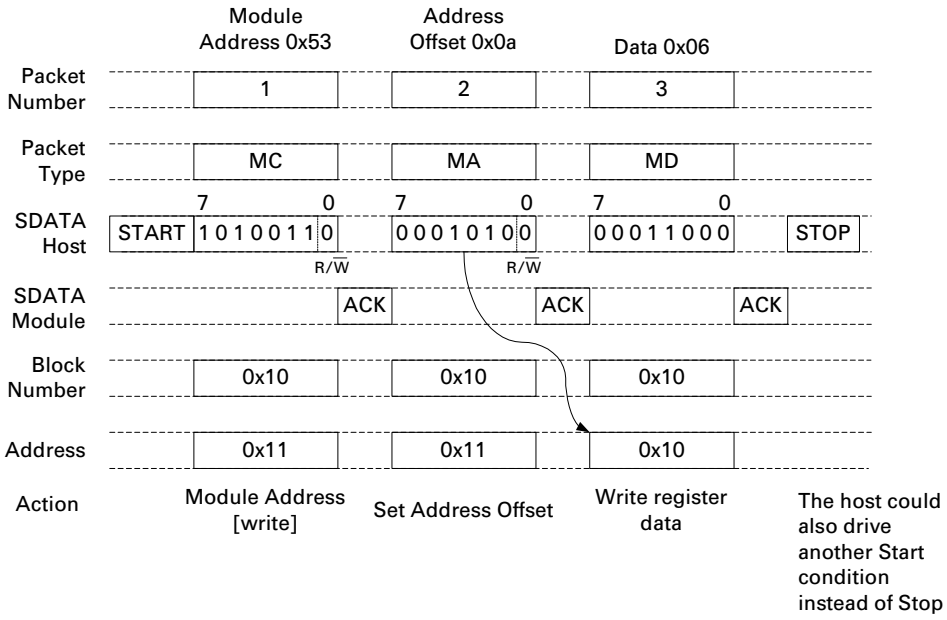



**Figure 35** Single byte write to the sensor registers

| Serial Control Data | Hex |
|---|---|
| Sent | a6, 14, 06 |
| Received | – |

**Reading from FWCOL:** In this example, the sensor reads the default value (0x01) from the FWCOL register, address 0x080a. This register is in block 0x10, so if example 4 was complete, the block number is already set correctly.

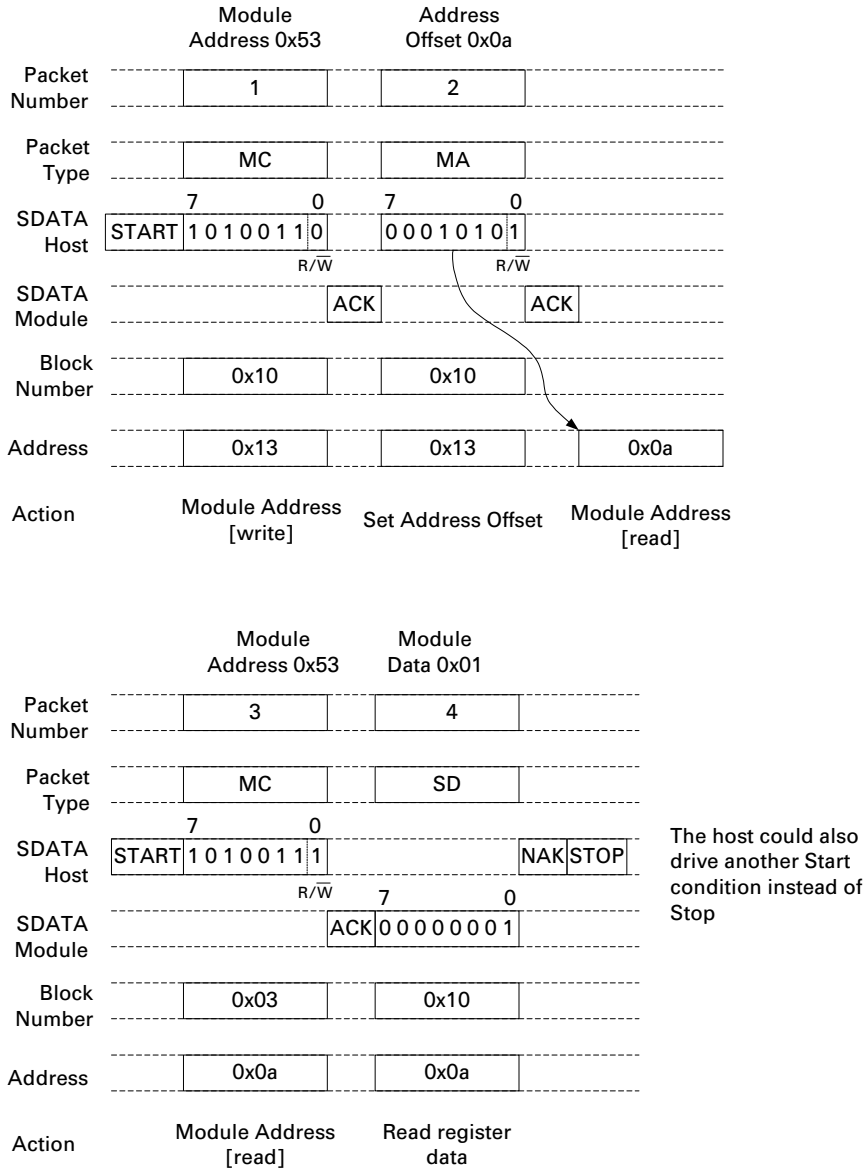



**Figure 36** Single byte read from sensor registers

| Serial Control Data | Hex |
|---|---|
| Sent | a6, 15, a7 |
| Received | 01 |

# 4 Parallel Data Output Protocols



## Protocol

Protocol is defined as the handshake or control signals that are output along with the 8-bit data bus that allow the host receiving the data to know how and when to read the data bus.

It is important to note that protocol and format are mostly independent, i.e. any format can be used with any protocol. The only exception is the Embedded Sychronization (ESYNC) mode can only be used with formats YCbYCr, CbYCrY, YCrYCb or CrYCbY.

## Parallel Output

The type of output is controlled by bit 0 of the SERIAL_CTRL register. The default output type is parallel.

**Serial Output Control**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SERIAL_CTRL | 0x108e | Serial control – serial data protocol, data order, synchronization controls | 0x0000 | 385 |

## Output Lines

The following lines are used for parallel output:

| Line | Description |
|---|---|
| VCLK | Output data clock |
| HSYNC (EOL) | Horizontal synchronization (End of Line) |
| VSYNC (EOF) | Vertical synchronization (End of Frame) |
| D0 | Data line 0 |
| D1 | Data line 1 |
| D2 | Data line 2 |
| D3 | Data line 3 |
| D4 | Data line 4 |
| D5 | Data line 5 |
| D6 | Data line 6 |
| D7 | Data line 7 |

# VCLK Frequency

The following registers directly control the VCLK frequency. Many of these registers are written by the simple control register:

**Expert registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CLK_DIV | 0x0080 | Clock dividers for top level | 0x0001 | 166 |
| IP_CLK_DIV | 0x0086 | Image pipeline clock dividers | 0x0000 | 169 |
| PARALLEL_CTRL_V | 0x011a | Parallel output control | 0x0003 | 199 |
| PARALLEL_CTRL_S | 0x013a | Parallel output control | 0x0000 | 217 |

The VCLK frequency varies with SSIZE, VSIZE, SPREF, VPREF, SSUB and VSUB fields of the SIZE register (address 0x0008), SOUT and VOUT in the OUTPUT_STYLE register (address 0x000a), the FRAME_RATE register (address 0x0012) and the input clock frequency (MCLK).

The simple control register adjusts VCLK for the lowest possible frequency for the lowest power consumption. Generally, increasing the frequency of VCLK will not affect functionality but will increase power and there will be more idle time between lines.

# Operating Parameters Registers

The following registers control the output protocol of the ADCM-2700 camera module:

**Operating parameters registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control:<br>Bit 8: Free running VCLK (FRVCLK)<br>Bit 9: EOF/EOL protocol (HSMODE)<br>Bit 10: 8-bit vs. 4-bit CCIR<br>Bit 11: Embedded sychronization | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control:<br>Bit 8: Free running VCLK (FRVCLK)<br>Bit 9: EOF/EOL protocol (HSMODE)<br>Bit 10: 8-bit vs. 4-bit CCIR<br>Bit 11: Embedded sychronization | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control:<br>Bit 8: Free running VCLK (FRVCLK)<br>Bit 9: EOF/EOL protocol (HSMODE)<br>Bit 10: 8-bit vs. 4-bit CCIR<br>Bit 11: Embedded sychronization | 0x9019 | 338 |
| PARALLEL_CTRL_V | 0x011a | Parallel output control, video mode:<br>Bit 14: Output line pacing<br>Bit 15: Output line pacing adjustment | 0x0003 | 199 |
| PARALLEL_CTRL_S | 0x013a | Parallel output control, still mode:<br>Bit 14: Output line pacing<br>Bit 15: Output line pacing adjustment | 0x0000 | 217 |

**Expert registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CCIR_TIMING | 0x1010 | CCIR timing control<br>VSYNC setup and hold | 0x0000 | 343 |
| CCIR_TIMING_2 | 0x1086 | CCIR timing control 2<br>HSYNC setup and data hold | 0x0000 | 381 |
| CCIR_TIMING_3 | 0x108a | CCIR timing control 3<br>Minimum HSYNC passive time | 0x0010 | 382 |
| SERIAL_CTRL | 0x108e | Serial control<br>Serial data protocol, data order, synchronization controls | 0x0000 | 385 |

# Parallel Output Protocols

The output protocol is controlled by two bits, FRVCLK and HSMODE.

- FRVCLK is bit 8 and HSMODE is bit 9 of the OUTPUT_CTRL_V, OUTPUT_CTRL_S or OUTPUT_CTRL registers
- FRVCLK controls the VLCK, there are two modes, free running and gated
- HSMODE controls the horizontal sychronization mode; either HSYNC high for the entire line with VSYNC high for the entire frame, or HSYNC high only for the last pixel of the line with VSYNC high for the last pixel of the frame
- VCLK, HSYNC and VSYNC polarities are controlled by bits 5 to 7 of OUTPUT_CTRL_V, OUTPUT_CTRL_S or OUTPUT_CTRL registers

## Example Syntax

In the following four output protocols, the data is from an $m$ by $n$ picture.

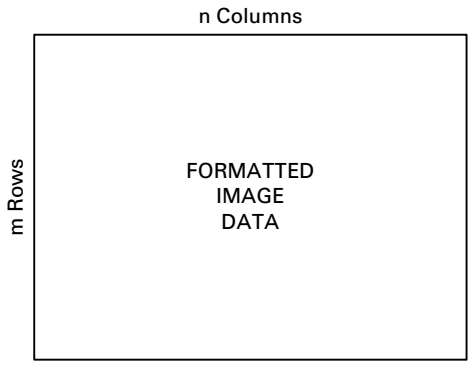



**Figure 37** Data input

In the following figures, data bytes in each line are represented by *Dmi*, where $m$ is the row number and $i$ is the data for the currently selected format. With formatting, there are between one and three bytes per pixel.

## Protocol 1a: Free Running VCLK, HSYNC/VSYNC, 8-Bit Data

FRVCLK = 0 HSMODE = 0

### Default Protocol Used for CCIR

The three control lines are VCLK, VSYNC and HSYNC, with VCLK running continuously:

- The data bus is valid on the rising edge of VCLK when both VSYNC and HSYNC are high
- VSYNC is high for an entire frame and low between frames
- HSYNC is high for an entire line and low between lines.
- The data on the data bus between lines and frames is 0x00

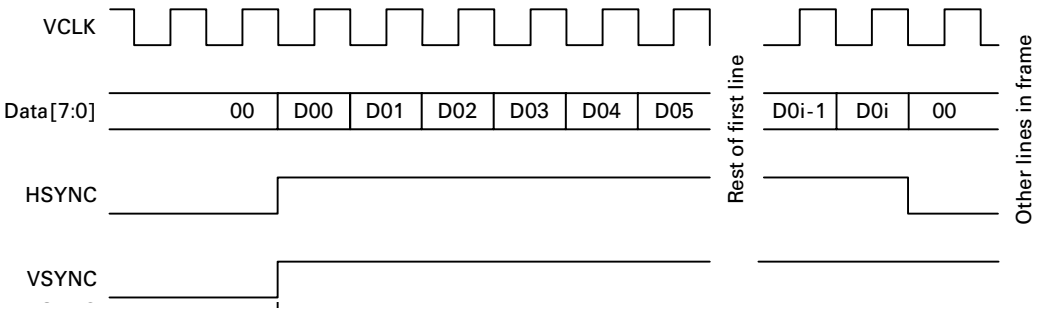



**Figure 38** Beginning of frame, protocol 1, default polarities

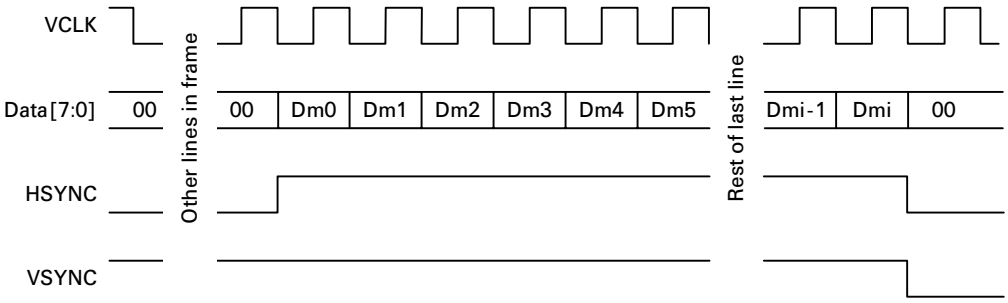



**Figure 39** End of frame, protocol 1, default polarities

The default protocol is mostly compatible with CCIR 656. To be fully compatible, the HSYNC and VSYNC information would have to be embedded in the data (see “CCIR 656 Embedded Synchronization" on page 115).

Since VCLK runs continuously and HSYNC stays high for an entire line, an entire line bursts out without interruption. The output then pauses until another full line is ready. The length of the pause between lines varies with configuration but is constant, within a few VCLKs within a frame.

Since the simple control register configures the VCLK clock frequency to be as slow as possible, the pause between lines is short relative to the length of the line. Simple control register settings can be overridden to increase the frequency of VCLK, which forces more VCLKs between each line.

Any format can be used with this protocol, but if CCIR 656 compatibility is desired, select the CbYCrY format (which is also the default).

## Protocol 1b: Free Running VCLK, HSYNC/VSYNC, 4-Bit Data

FRVCLK = 0 HSMODE = 0

### CCIR protocol output on lower four data lines, upper four bits = 0

The three control lines are VCLK, VSYNC and HSYNC, with VCLK running continuously:

- The data bus is valid on the rising edge of VCLK when both VSYNC and HSYNC are high
- VSYNC is high for an entire frame and low between frames
- HSYNC is high for an entire line and low between lines.
- The data on the data bus between lines and frames is 0x00
- Data nibble order is controlled by the SDO bit in SERAIL_CTRL.
- This mode is selected with the 4BCCIR bit in OUTPUT_CTRL

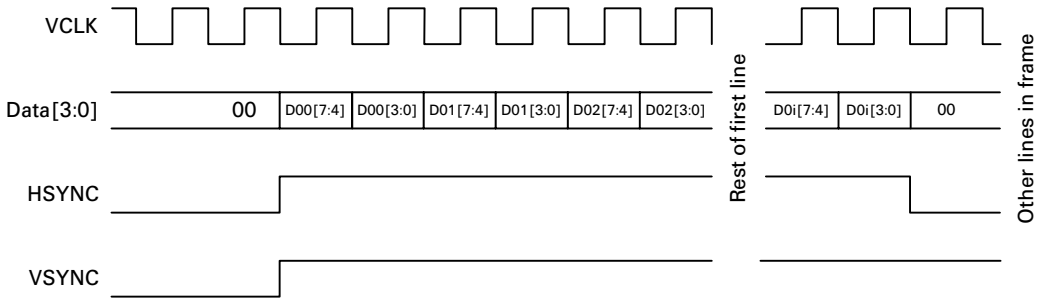



**Figure 40** Beginning of frame, protocol 1b, default polarities, most significant nibble first.

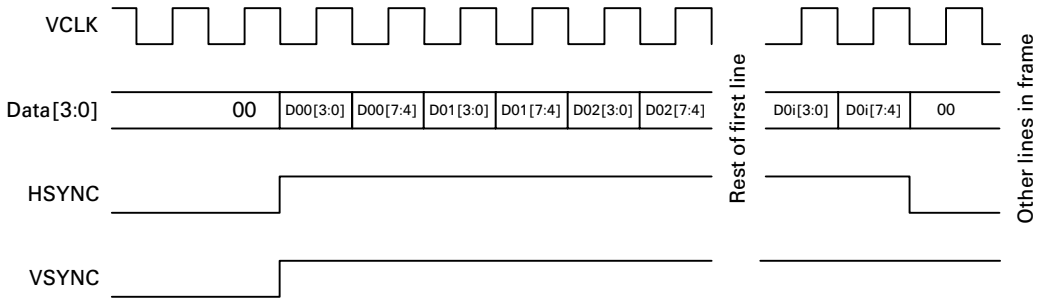



**Figure 41** Beginning of frame, protocol 1b, default polarities, least significant nibble first.

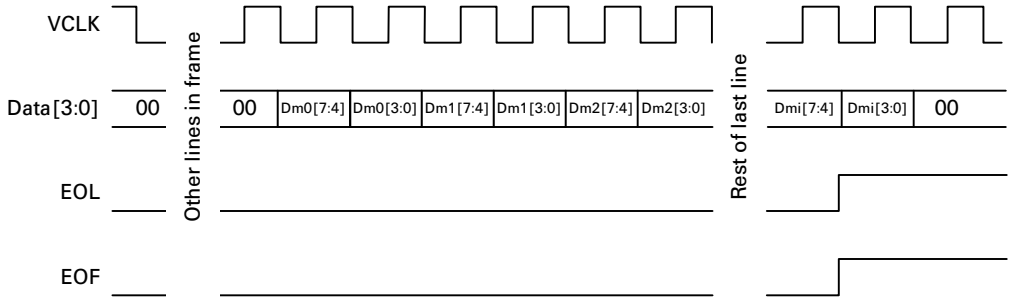



**Figure 42** End of frame, protocol 1b, default polarities, most significant nibble first

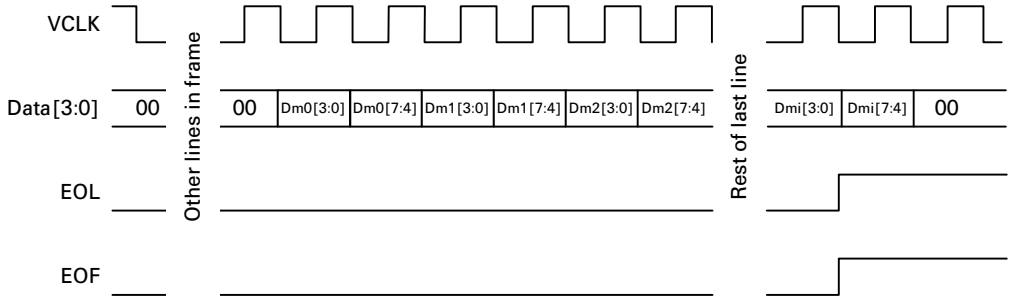



**Figure 43** End of frame, protocol 1b, default polarities, least significant nibble first

## Protocol 2: Gated VCLK, HSYNC / VSYNC

FRVCLK = 1 HSMODE = 0

The three control lines are VCLK, VSYNC and HSYNC:

- VCLK stops between lines and frames
- The data bus is valid on the rising edge of VCLK when both VSYNC and HSYNC are high
- VSYNC is high for an entire frame and low between frames
- HSYNC is high for an entire line and low between lines
- The data from the last pixel in a row will be kept on the data bus between lines and frames

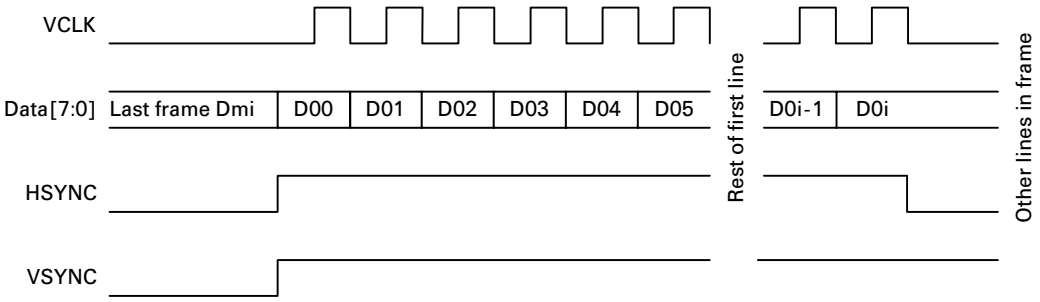



**Figure 44** Beginning of frame, protocol 2, default polarities

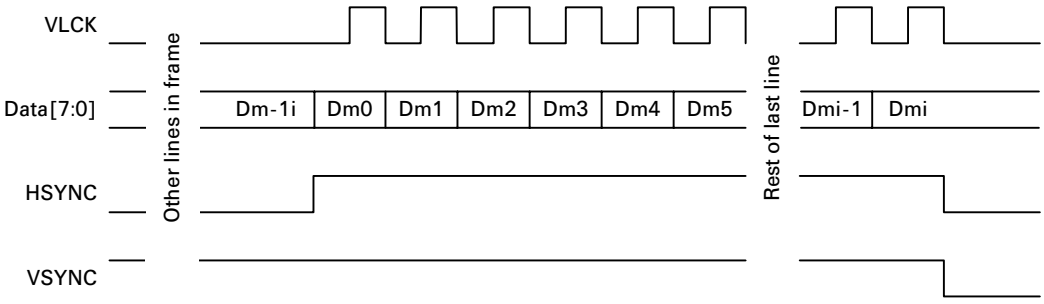



**Figure 45** End of frame, protocol 2, default polarities

Since VCLK runs continuously within a line and HSYNC stays high for an entire line, an entire line bursts out without interruption. The output then pauses until another full line is ready. The length of the pause between lines varies with configuration but is constant, within a few VCLKs within a frame.

## Protocol 3: Gated VCLK, EOL / EOF

FRVCLK = 1 HSMODE = 1

The three control lines are VCLK, EOL and EOF:

- The EOL line is physically the HSYNC line, the EOF the VSYNC line
- VCLK stops between lines and frames
- The data bus is valid on the rising edge of VCLK. EOL will only be high for the last pixel in a line and between lines
- EOF will be high for the last pixel in a frame. Both EOL and EOF will be high between frames
- The data from the last pixel in a row will be kept on the data bus between lines and frames

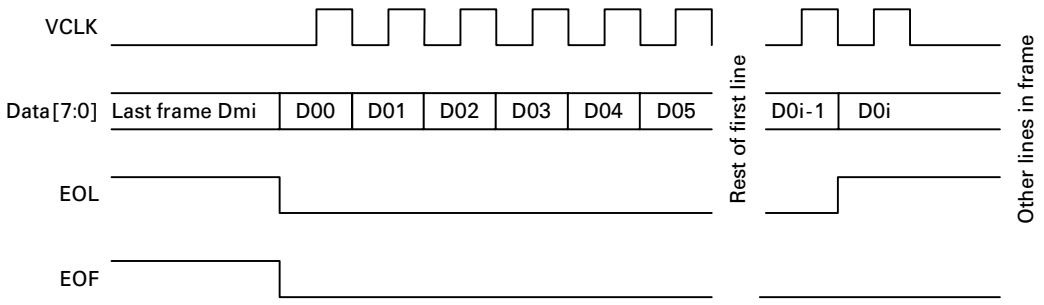



**Figure 46** Beginning of frame, protocol 3, default polarities.

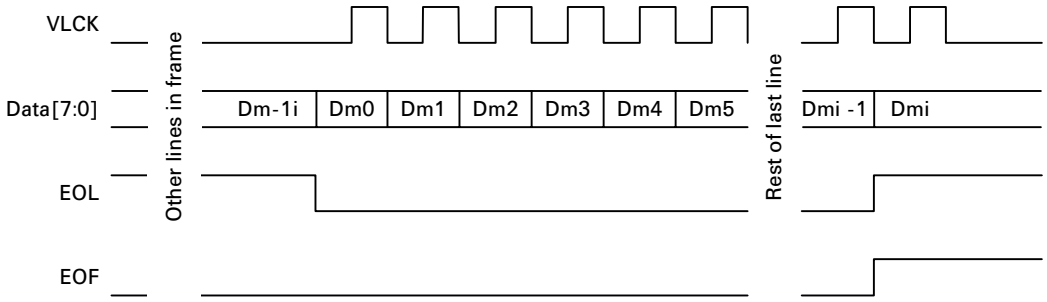



**Figure 47** End of frame, protocol 3, default polarities.

## Protocol 4: Free Running VCLK, EOL / EOF

FRVCLK = 0 HSMODE = 1

### Not Recommended

The three control lines are VCLK, EOL and EOF:

- The EOL line is physically the HSYNC line; the EOF for the VSYNC line
- EOL will only be high for the last pixel in a line and low between lines
- EOF will be high for the last pixel in a frame
- Both EOL and EOF will be high between frames
- DATA, EOL and EOF changes happen on the rising edge of VCLK and they are stable on the falling edge
- The data on the data bus between lines and frames is 0x00

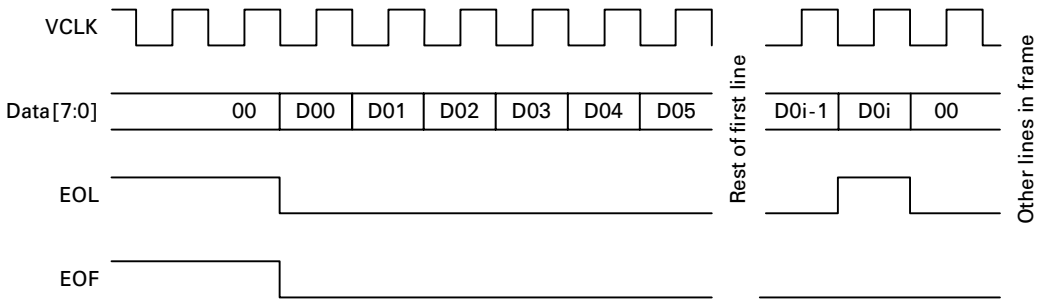



**Figure 48** Beginning of frame, protocol 4, default polarities

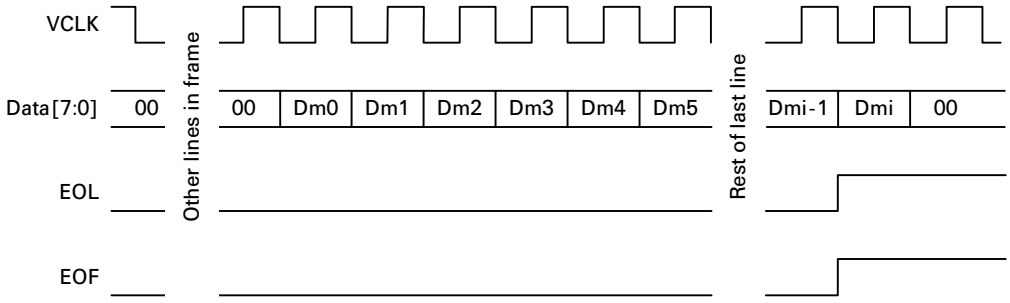



**Figure 49** End of frame, protocol 4, default polarities

# CCIR Fine Tuning

When outputting data with protocol 1, using CCIR formatting, the timing can be adjusted with the following registers:

**CCIR fine tuning registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| CCIR_TIMING | 0x1010 | CCIR timing control<br>VSYNC setup and hold, number of dummy HSYNCs | 0x0000 | 343 |
| CCIR_TIMING_2 | 0x1086 | CCIR timing control 2<br>HSYNC setup and data hold | 0x0000 | 381 |
| CCIR_TIMING_3 | 0x108a | CCIR timing control 3<br>Minimum number of VCLK's in HSYNC | 0x0010 | 382 |

## CCIR_TIMING Register

The CCIR_TIMING register (address 0x1010, see page 343) has three parameters:

- VSYNC setup time before HSYNC assertion
- VSYNC hold time after HSYNC deassertion
- Number of dummy HSYNCs inserted after a frame

### VSYNC Setup

The V_SETUP variable is the number of VCLKs cycles that the VSYNC line is asserted before the HSYNC line is asserted. The range is from 0 to 63 and the default value is 0.

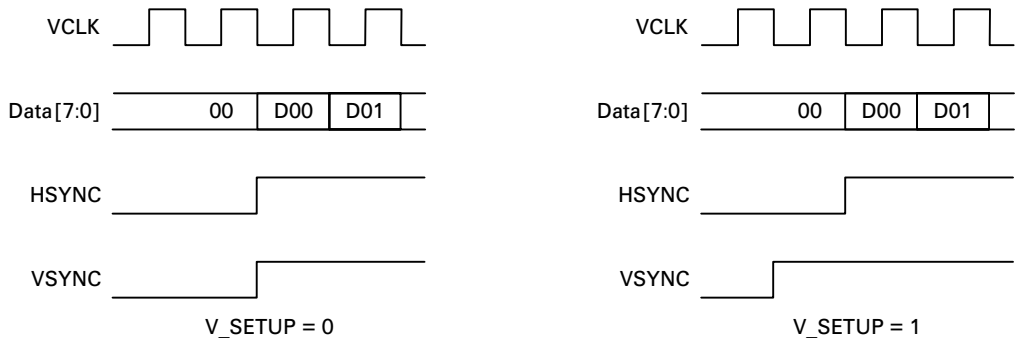



**Figure 50** V_SETUP example, beginning of frame

### VSYNC Hold

The V_HOLD variable is the number of VCLKs that the VSYNC line stays asserted after the HSYNC line is deasserted. The range is from 0 to 63 and the default value is 0.

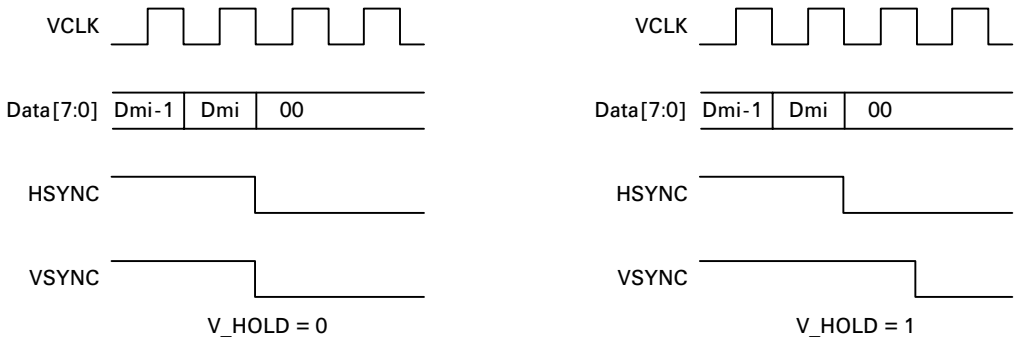



**Figure 51** V_HOLD example, end of frame

### Dummy HSYNCs

The D_HSYNC variable is the number of dummy HSYNCs inserted after the data frame. The range is from 0 to 15 and the default number is 0. *LL -1* in the figure stands for the next to last line.

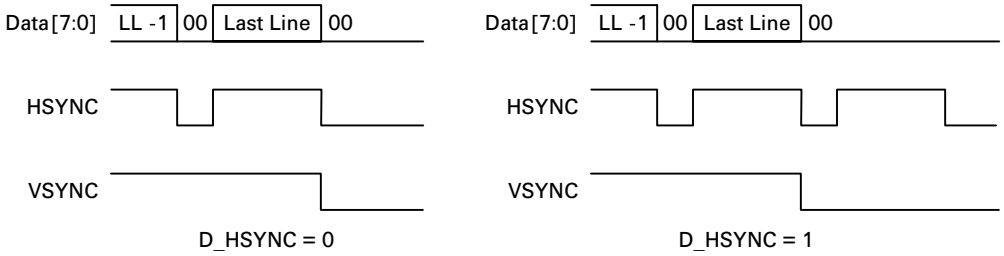



**Figure 52** D_HSYNC example, end of frame

## CCIR_TIMING2 Register

The CCIR_TIMING2 register (address 0x1086, see page 381) has two parameters:

- HSYNC setup time before data deassertion
- HSYNC hold time after data assertion

### HSYNC Setup

The HSDS variable in the CCIR_TIMING2 register is the number of VCLKs cycles that the HSYNC line is asserted before the data is asserted. The range is from 0 to 63 and the default value is 0. If the value is non-zero, the VSYNC edge also moves in tandem with the HSYNC edge.

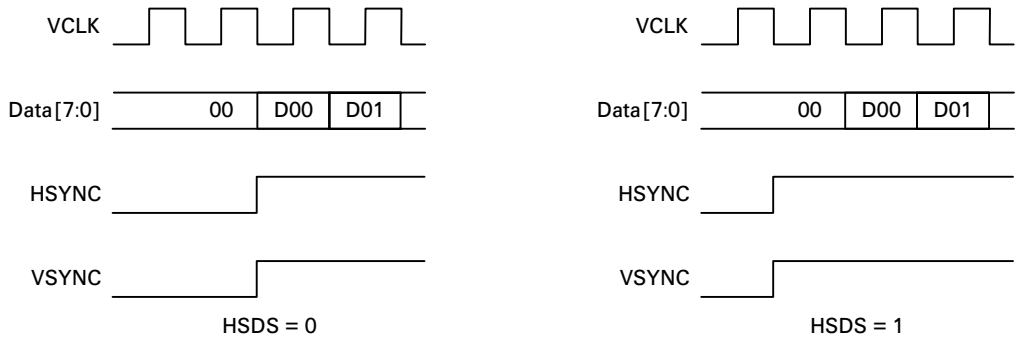



**Figure 53** HSDS example, beginning of frame

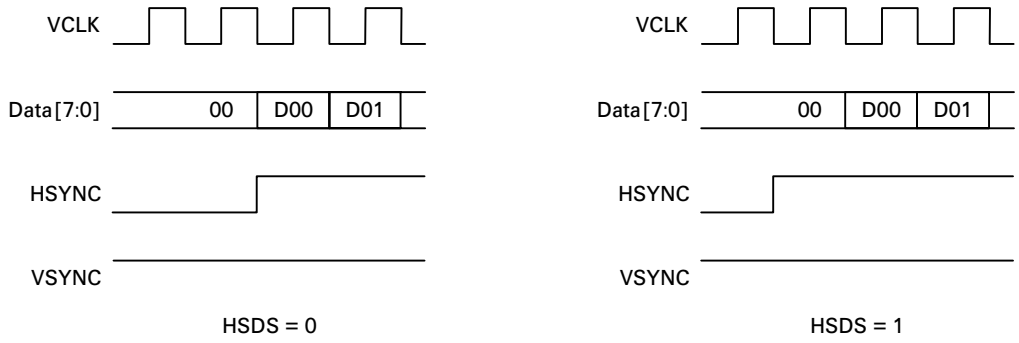



**Figure 54** HSDS example, middle of frame

### HSYNC Hold

The HSDH variable in the CCIR_TIMING2 register is the number of VCLKs that the HSYNC line stays asserted after the data is deasserted. The range is from 0 to 63 and the default value is 0. If the value is non-zero, the VSYNC at the end of the frame will also move in tandem with HSYNC.

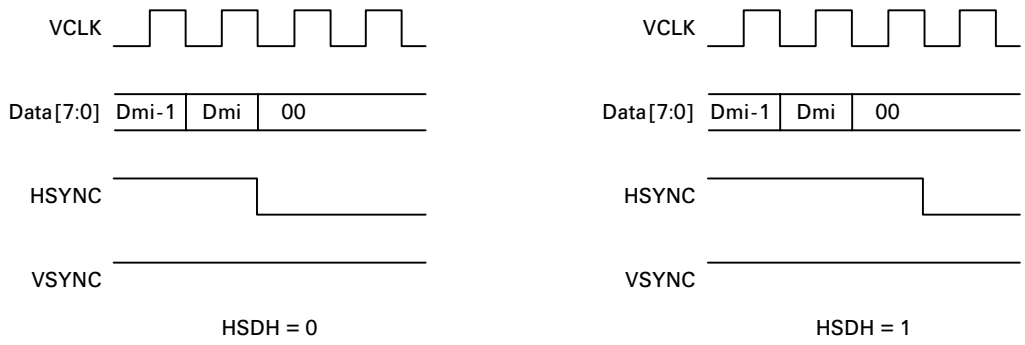



**Figure 55** HSDH example, middle of frame

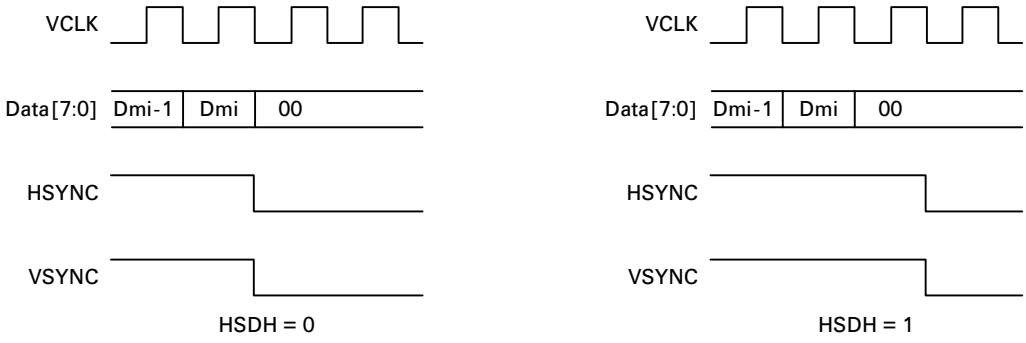



**Figure 56** HSDH example, end of frame

## CCIR_TIMING3 Register

The CCIR_TIMING3 register (address 0x1088, see page 382) has two parameters:

- Minimum HSYNC passive time
- A flag to disable the usage of HSYNC to signal truncated frames

### Minimum HSYNC Passive Time

The MHSPT variable in the CCIR_TIMING3 register is the minimum number of VCLKs cycles that the HSYNC line is deasserted between frames. This is a minimum number, the actual HSYNC passive time may be greater and it is somewhat dependent on the window size, frame rate and clock speed. The range is from 0 to 63 and the default value is 0x10, 16 decimal.

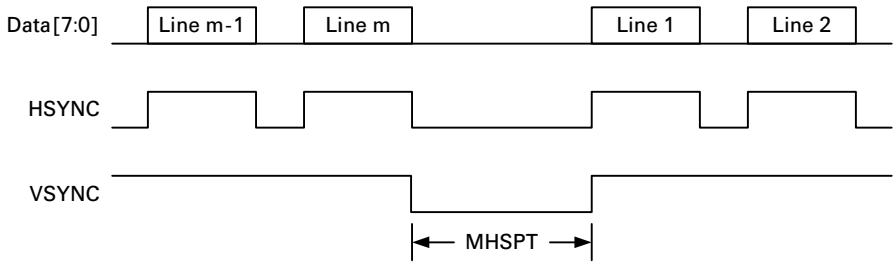



**Figure 57** Minimum HSYNC passive time example

# Inverting Polarities

## VCLK, HSYNC and VSYNC Polarities

The output polarity of the VCLK, HSYNC and VSYNC lines are controlled by the OUTPUT_CTRL_V and OUTPUT_CTRL_S registers. The OUTPUT_CTRL register is written to by the simple control register from the relevant video or still register.

**Operating parameters registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control:<br>Bit 5: VCLK polarity<br>Bit 6: VSYNC polarity<br>Bit 7: HSYNC polarity | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control:<br>Bit 5: VCLK polarity<br>Bit 6: VSYNC polarity<br>Bit 7: HSYNC polarity | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control:<br>Bit 5: VCLK polarity<br>Bit 6: VSYNC polarity<br>Bit 7: HSYNC polarity | 0x9019 | 338 |

If the VCLK polarity is inverted, then the data changes on the rising edge of VCLK and data is stable and should be read on the falling edge.

Similarly, setting the polarity bits for HSYNC or VSYNC will cause those signals to be inverted from their normal levels.

## EOF and EOL Polarities

When the camera module is using one of the EOL/EOF modes, (protocols 3 and 4), the EOL and EOF polarities are the inverse of the HSYNC and VSYNC polarities.

## HSYNC Line Pacing

When the camera module is used with the full size array or with the sizer set to an even multiple such as 1/2 or 1/4, the HSYNC line toggles at a fixed frequency. If the sizer is used with an odd multiple, such as 2/3, then the HSYNC line will toggle unevenly.

**HSYNC pacing control**

| Register name | Address | Control Bits | Default | Page |
|---|---|---|---|---|
| PARALLEL_CTRL_V | 0x011a | DOLP (bit 14), disable output line pacing<br>DOLPA (bit 15), disable line pacing adjustments | 0x0003 | 199 |
| PARALLEL_CTRL_S | 0x013a | DOLP (bit 14), disable output line pacing<br>DOLPA (bit 15), disable line pacing adjustments | 0x0000 | 217 |
| PARALLEL_CTRL | 0x100a | DOLP (bit 14), disable output line pacing<br>DOLPA (bit 15), disable line pacing adjustments | 0x0000 | 340 |

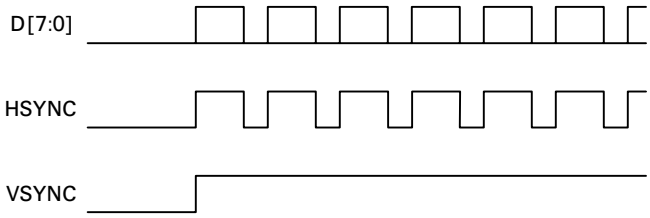



**Figure 58** Beginning of frame, sizer disabled, line pacing enabled or disabled

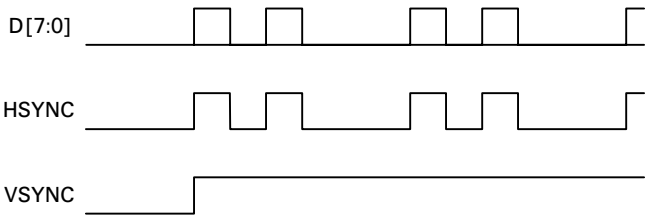



**Figure 59** Beginning of frame, sizer set to 2/3, line pacing disabled

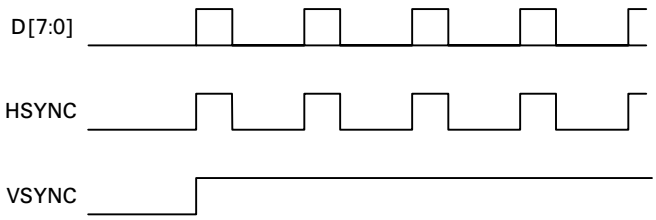



**Figure 60** Beginning of frame, sizer set to 2/3, line pacing enabled

# Default CCIR Timing

If the CCIR timing registers are set to their default values, the following timing diagrams apply. The value of $t_{VCLK}$ is dependent on the input clock, MCLK and the clock divisors.

## Data to VCLK Timing

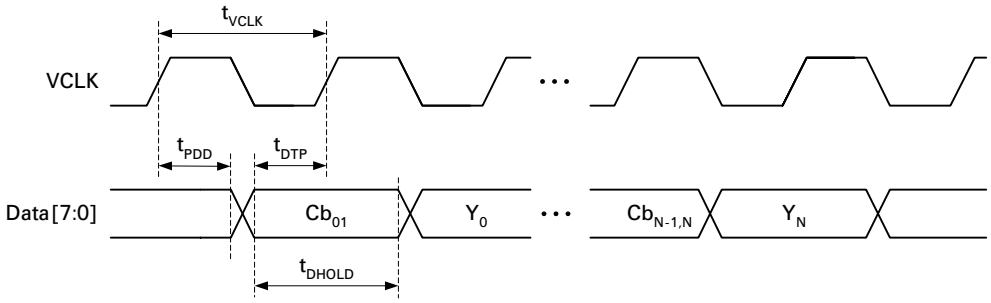



**Figure 61** Data to VCLK timing

| | | Value | | | |
|---|---|---|---|---|---|
| **Parameter** | **Symbol** | **Minimum** | **Typical** | **Maximum** | **Units** |
| VCLK to data valid | $t_{PDD}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| Valid data to VCLK | $t_{DTP}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| Data hold time | $t_{DHOLD}$ | $t_{VCLK} - 12$ | $t_{VCLK} - 6$ | $t_{VCLK}$ | ns |

## HSYNC Timing

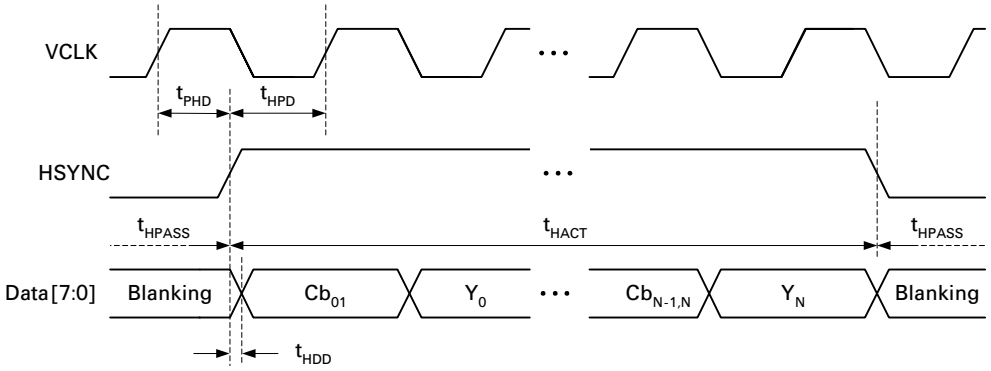



**Figure 62** HSYNC timing

| | | Value | | | |
|---|---|---|---|---|---|
| **Parameter** | **Symbol** | **Minimum** | **Typical** | **Maximum** | **Units** |
| VCLK to HSYNC delay | $t_{PHD}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| HSYNC to VCLK delay | $t_{HPD}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| HSYNC active time (high) | $t_{HACT}$ | 2 x image width | 2 x image width | 2 x image width | VCLKs |
| HSYNC passive time (low) | $t_{HPASS}$ | 16 | >> 16 | | VCLKs |
| Data to HSYNC jitter | $t_{HDD}$ | – 6 | 0 | 6 | ns |

## VSYNC Timing 1

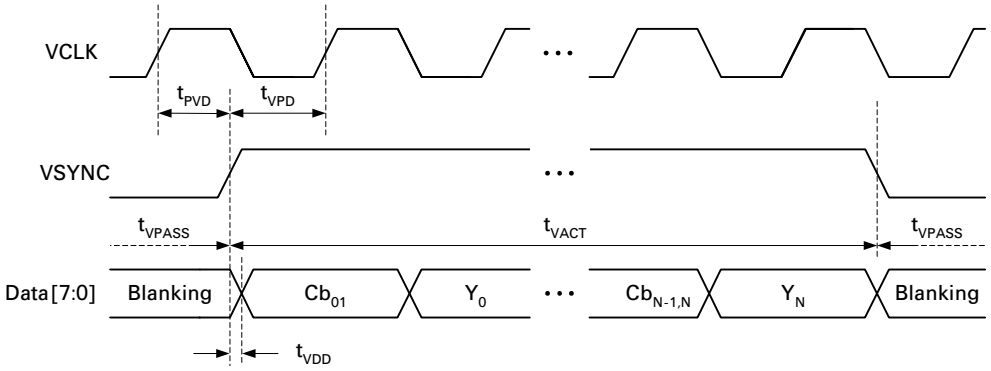



**Figure 63** VSYNC timing 1

| | | | Value | | |
|---|---|---|---|---|---|
| **Parameter** | **Symbol** | **Minimum** | **Typical** | **Maximum** | **Units** |
| VCLK to VSYNC delay | $t_{PVD}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| VSYNC to VCLK delay | $t_{VPD}$ | $t_{VCLK}/2 - 6$ | $t_{VCLK}/2$ | $t_{VCLK}/2 + 6$ | ns |
| VSYNC active time (high) | $t_{VACT}$ | ((2 x image_width)+16) x image_height -16 | >> min | | VCLKs |
| VSYNC passive time (low) | $t_{VPASS}$ | 16 | >> 16 | | VCLKs |
| Data to VSYNC jitter | $t_{VDD}$ | – 6 | 0 | 6 | ns |

## VSYNC Timing 2

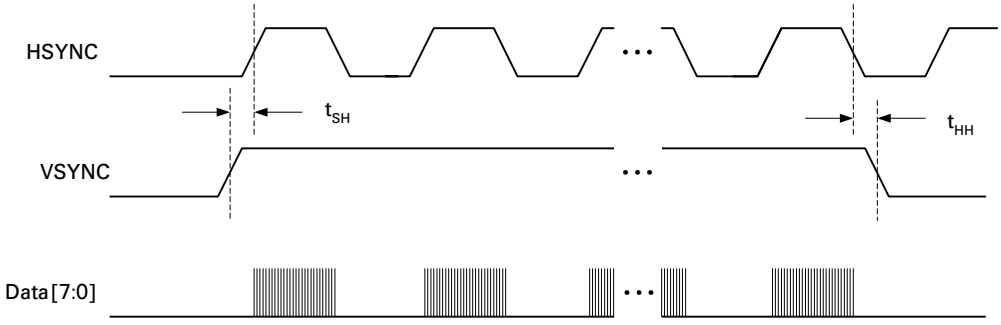



**Figure 64** VSYNC timing 2

| | | Value | | | |
|---|---|---|---|---|---|
| **Parameter** | **Symbol** | **Minimum** | **Typical** | **Maximum** | **Units** |
| VSYNC setup time to HSYNC | $t_{SH}$ | 0 | 0 | 0 | ns |
| VSYNC hold time to HSYNC | $t_{HH}$ | 0 | 0 | 0 | ns |

# 5 Data Output Formats



The ADCM-2700 camera module outputs video or still images over a parallel port (eleven lines, eight bits of data and three protocol signals) or a synchronous serial port (four lines, clock, data and two optional handshake lines). The camera module is controlled using a serial control bus (see “Serial Control Interface" on page 67) that is used to read and write registers.

Details of the serial and parallel output protocols are detailed in Chapter 4, “Parallel Data Output Protocols,” starting on page 83 and in Chapter 6, “Synchronous Serial Output Protocols,” starting on page 119.

## Data Formats

Format is defined as the content of the data, i.e., what the data represents. For example, the data could be 24-bit RGB with the red byte first, followed by green and then blue. Or it could be YCbCr 4:2:2 in the order of Cb, then Y, then Cr and then Y. Both examples are described in more detail in this chapter.

### Data Order

Image data is output in raster order, one line at a time, from top to bottom, with each line output one pixel at a time from left to right.

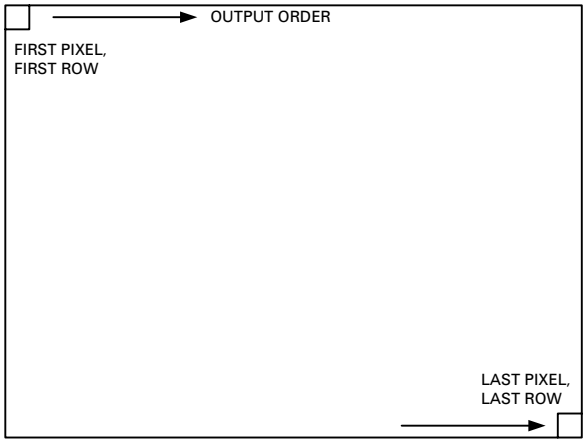



**Figure 65** Data output order

## Data Format Registers

The data format is controlled by the following registers:

**Simple control register**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_STYLE | 0x000a | Bits [3:0] control the video output format: corresponds to OUTPUT_CTRL_V[3:0] | 0x0909 | 147 |
| | | Bits [11:8] control the still output format: corresponds to OUTPUT_CTRL_S[3:0] | | |

**Expert camera controller registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control (working copy) | 0x9019 | 338 |

### Quick Review of the Simple Control Registers

Simple Control Registers (SCR) were created to reduce some of the complexity in the many control registers of the ADCM-2700. By setting a few simple control registers (addresses 0x0000 to 0x007e) and then setting the CONFIG bit in the CONTROL register (address 0x0002, see page 140), many different expert registers (0x0080 to 0x00ff and 0x0800 to 0x1800) and operating parameters (0x0100 to 0x07ff) are set to the correct values to achieve the functions requested in simple control register. Once simple control register CONFIG is complete, further modifications of expert registers and operating parameters can be made.

### Using the Simple Control Register

To use the simple control register, the following procedure must be used:

1. Clear the RUN bit in the CONFIG register.
2. Set the desired parameters into the simple control registers.
3. Set the CONFIG bit in the CONFIG register. This bit will autoclear when the configuration is complete.
4. Wait for the CONFIG bit to clear.
5. Set the RUN bit in CONFIG.

## OUTPUT_STYLE Register

The OUTPUT_STYLE register (address 0x000a, see page 147) defines the output formats for the video and still modes. Bits [3:0] define the video output format, bits [11:8] define the still.

When the CONFIG bit in the CONTROL register (address 0x0002, see page 140) is set, these two sets of four bits are copied to OUTPUT_CTRL_V[3:0] (address 0x0110, see page 192) and OUTPUT_CTRL_S[3:0] (address 0x0130, see page 210).

Once CONFIG is complete, OUTPUT_CTRL_V and OUTPUT_CTRL_S can be further modified, if necessary, before setting the RUN bit in the CONTROL register. When either the RUN or SNAP bits are set, the corresponding register is copied to the OUTPUT_CTRL register (address 0x1008, see page 338), which is the actual working copy.

## OUTPUT_CTRL_V [3:0] / OUTPUT_CTRL_S [3:0]

These two registers behave identically except that one is used for video mode and the other is used for still mode. Bits [3:0] define the output to be one of 16 possible formats.

## OUTPUT_CTRL [3:0]

The OUTPUT_CTRL register is the current working copy of either the video or still register, depending on which mode the camera module is currently operating in.

**Available output formats**

| Bits[3:0] | Format | Bits/ Pixel | Description |
|---|---|---|---|
| 0 | 888 RGB | 24 | Each 24-bit RGB pixel is output in three bytes in the following order: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| 2 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |
| 3 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

Repeat for the next pixel

| Bits[3:0] | Format | Bits/ Pixel | Description |
|---|---|---|---|
| 1 | 666A RGB | 18 | Each 24-bit RGB pixel has the lower two bits of each color truncated. The remaining 18 bits are packed into bytes and output. Four pixels (72 bits) are packed into nine bytes. The output order of the nine bytes for each set of four pixels is: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $R_1$7 | $R_1$6 | $R_1$5 | $R_1$4 | $R_1$3 | $R_1$2 | $G_1$7 | $G_1$6 |
| 2 | $G_1$5 | $G_1$4 | $G_1$3 | $G_1$2 | $B_1$7 | $B_1$6 | $B_1$5 | $B_1$4 |
| 3 | $B_1$3 | $B_1$2 | $R_2$7 | $R_2$6 | $R_2$5 | $R_2$4 | $R_2$3 | $R_2$2 |
| 4 | $G_2$7 | $G_2$6 | $G_2$5 | $G_2$4 | $G_2$3 | $G_2$2 | $B_2$7 | $B_2$6 |
| 5 | $B_2$5 | $B_2$4 | $B_2$3 | $B_2$2 | $R_3$7 | $R_3$6 | $R_3$5 | $R_3$4 |
| 6 | $R_3$3 | $R_3$2 | $G_3$7 | $G_3$6 | $G_3$5 | $G_3$4 | $G_3$3 | $G_3$2 |
| 7 | $B_3$7 | $B_3$6 | $B_3$5 | $B_3$4 | $B_3$3 | $B_3$2 | $R_4$7 | $R_4$6 |
| 8 | $R_4$5 | $R_4$4 | $R_4$3 | $R_4$2 | $G_4$7 | $G_4$6 | $G_4$5 | $G_4$4 |
| 9 | $G_4$3 | $G_4$2 | $B_4$7 | $B_4$6 | $B_4$5 | $B_4$4 | $B_4$3 | $B_4$2 |

Repeat for the next four pixels. If the output image width is not a multiple of four, the last column will be repeated until a multiple of four is reached.

**Available output formats (continued)**

| Bits[3:0] | Format | Bits/ Pixel | Description |
|---|---|---|---|
| 2 | 666B RGB | 18 | Each 24-bit RGB pixel has the lower two bits of each color truncated. The pixel order is the same as 888 RGB. The JUST bit controls the left or right justification: |

JUST bit = 1

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | R7 | R6 | R5 | R4 | R3 | R2 |
| 2 | 0 | 0 | G7 | G6 | G5 | G4 | G3 | G2 |
| 3 | 0 | 0 | B7 | B6 | B5 | B4 | B3 | B2 |

JUST bit = 0

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | R7 | R6 | R5 | R4 | R3 | R2 | 0 | 0 |
| 2 | G7 | G6 | G5 | G4 | G3 | G2 | 0 | 0 |
| 3 | B7 | B6 | B5 | B4 | B3 | B2 | 0 | 0 |

Repeat for the next pixel

| | | | |
|---|---|---|---|
| 3 | 565 RGB | 16 | Each 24-bit RGB pixel has the lower two bits of green truncated. Red and blue have the lower three bits truncated. The pixel is packed into two bytes: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | R7 | R6 | R5 | R4 | R3 | G7 | G6 | G5 |
| 2 | G4 | G3 | G2 | B7 | B6 | B5 | B4 | B3 |

Repeat for the next pixel

| | | | |
|---|---|---|---|
| 4 | 444A RGB | 12 | Each 24-bit RGB pixel has the lower four bits of each color truncated. The remaining 12 bits are packed into bytes and output. Two pixels (24 bits) are packed into three bytes: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $R_1$7 | $R_1$6 | $R_1$5 | $R_1$4 | $G_1$7 | $G_1$6 | $G_1$5 | $G_1$4 |
| 2 | $B_1$7 | $B_1$6 | $B_1$5 | $B_1$4 | $R_2$7 | $R_2$6 | $R_2$5 | $R_2$4 |
| 3 | $G_1$7 | $G_1$6 | $G_2$5 | $G_2$4 | $B_2$7 | $B_2$6 | $B_2$5 | $B_2$4 |

Repeat for the next two pixels. If the output image width is not a multiple of two, the last column will be replicated

**Available output formats (continued)**

| Bits[3:0] | Format | Bits/ Pixel | Description | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 444B RGB | 12 | Each 24-bit RGB pixel has the lower four bits of each color truncated. The remaining 12 bits and four padding zeros are packed into two bytes. The JUST bit controls the left or right justification: | | | | | | | |
| | JUST bit = 1 | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | 0 | 0 | 0 | 0 | R7 | R6 | R5 | R4 |
| | | | 2 | G7 | G6 | G5 | G4 | B7 | B6 | B5 | B4 |
| | JUST bit = 0 | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | R7 | R6 | R5 | R4 | G7 | G6 | G5 | G4 |
| | | | 2 | B7 | B6 | B5 | B4 | 0 | 0 | 0 | 0 |
| | | | Repeat for the next pixel | | | | | | | | |
| 6 | 444C RGB | 12 | Each 24-bit RGB pixel has the lower four bits of each color truncated. The pixel order is the same as 888 RGB. The JUST bit controls the left or right justification: | | | | | | | |
| | JUST bit = 1 | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | 0 | 0 | 0 | 0 | R7 | R6 | R5 | R4 |
| | | | 2 | 0 | 0 | 0 | 0 | G7 | G6 | G5 | G4 |
| | | | 3 | 0 | 0 | 0 | 0 | B7 | B6 | B5 | B4 |
| | JUST bit = 0 | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | R7 | R6 | R5 | R4 | 0 | 0 | 0 | 0 |
| | | | 2 | G7 | G6 | G5 | G4 | 0 | 0 | 0 | 0 |
| | | | 3 | B7 | B6 | B5 | B4 | 0 | 0 | 0 | 0 |
| | | | Repeat for the next pixel. | | | | | | | | |
| 7 | 332 RGB | 8 | Each 24-bit RGB pixel has the lower five bits of green and red truncated. Blue has the lower six bits truncated. The pixel is packed into one byte: | | | | | | | |
| | | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | R7 | R6 | R5 | G7 | G6 | G5 | B7 | B6 |
| | | | Repeat for the next pixel | | | | | | | | |

**Available output formats (continued)**

| Bits[3:0] | Format | Bits/ Pixel | Description |
|---|---|---|---|
| 8 | 4:2:2A YCbYCr | 16 | Each horizontally adjacent pair of 24-bit YCbCr pixels has the Cb and Cr values averaged. This averaged value is shown below as $Cb_{XY}$ or $Cr_{XY}$. If the two pixels have values of $Y_1$, $Cb_1$, $Cr_1$ and $Y_2$, $Cb_2$ $Cr_2$, the 4-byte output will be: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $Y_17$ | $Y_16$ | $Y_15$ | $Y_14$ | $Y_13$ | $Y_12$ | $Y_11$ | $Y_10$ |
| 2 | $Cb_{12}7$ | $Cb_{12}6$ | $Cb_{12}5$ | $Cb_{12}4$ | $Cb_{12}3$ | $Cb_{12}2$ | $Cb_{12}1$ | $Cb_{12}0$ |
| 3 | $Y_27$ | $Y_26$ | $Y_25$ | $Y_24$ | $Y_23$ | $Y_22$ | $Y_21$ | $Y_20$ |
| 4 | $Cr_{12}7$ | $Cr_{12}6$ | $Cr_{12}5$ | $Cr_{12}4$ | $Cr_{12}3$ | $Crb_{12}2$ | $Cr_{12}1$ | $Cr_{12}0$ |

Repeat for the next two pixels. If the output image width is not a multiple of two, the last column will be replicated.

| | | | |
|---|---|---|---|
| 9 | 4:2:2B CbYCrY | 16 | Similar to 4:2:2A YCbYCr except that the output order is different: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $Cb_{12}7$ | $Cb_{12}6$ | $Cb_{12}5$ | $Cb_{12}4$ | $Cb_{12}3$ | $Cb_{12}2$ | $Cb_{12}1$ | $Cb_{12}0$ |
| 2 | $Y_17$ | $Y_16$ | $Y_15$ | $Y_14$ | $Y_13$ | $Y_12$ | $Y_11$ | $Y_10$ |
| 3 | $Cr_{12}7$ | $Cr_{12}6$ | $Cr_{12}5$ | $Cr_{12}4$ | $Cr_{12}3$ | $Crb_{12}2$ | $Cr_{12}1$ | $Cr_{12}0$ |
| 4 | $Y_27$ | $Y_26$ | $Y_25$ | $Y_24$ | $Y_23$ | $Y_22$ | $Y_21$ | $Y_20$ |

Repeat for the next two pixels. If the output image width is not a multiple of two, the last column will be replicated.

| | | | |
|---|---|---|---|
| 10 | 4:2:2C YCrYCb | 16 | Similar to 4:2:2A YCbYCr except that the output order is different: |

| Byte # | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $Y_17$ | $Y_16$ | $Y_15$ | $Y_14$ | $Y_13$ | $Y_12$ | $Y_11$ | $Y_10$ |
| 2 | $Cr_{12}7$ | $Cr_{12}6$ | $Cr_{12}5$ | $Cr_{12}4$ | $Cr_{12}3$ | $Crb_{12}2$ | $Cr_{12}1$ | $Cr_{12}0$ |
| 3 | $Y_27$ | $Y_26$ | $Y_25$ | $Y_24$ | $Y_23$ | $Y_22$ | $Y_21$ | $Y_20$ |
| 4 | $Cb_{12}7$ | $Cb_{12}6$ | $Cb_{12}5$ | $Cb_{12}4$ | $Cb_{12}3$ | $Cb_{12}2$ | $Cb_{12}1$ | $Cb_{12}0$ |

Repeat for the next two pixels. If the output image width is not a multiple of two, the last column will be replicated.

**Available output formats (continued)**

| Bits[3:0] | Format | Bits/ Pixel | Description |
|---|---|---|---|
| 11 | 4:2:2D CrYCbY | 16 | Similar to 4:2:2A YCbYCr except that the output order is different: |

| **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | $Cr_{12}7$ | $Cr_{12}6$ | $Cr_{12}5$ | $Cr_{12}4$ | $Cr_{12}3$ | $Crb_{12}2$ | $Cr_{12}1$ | $Cr_{12}0$ |
| 2 | $Y_17$ | $Y_16$ | $Y_15$ | $Y_14$ | $Y_13$ | $Y_12$ | $Y_11$ | $Y_10$ |
| 3 | $Cb_{12}7$ | $Cb_{12}6$ | $Cb_{12}5$ | $Cb_{12}4$ | $Cb_{12}3$ | $Cb_{12}2$ | $Cb_{12}1$ | $Cb_{12}0$ |
| 4 | $Y_27$ | $Y_26$ | $Y_25$ | $Y_24$ | $Y_23$ | $Y_22$ | $Y_21$ | $Y_20$ |

Repeat for the next two pixels. If the output image width is not a multiple of two, the last column will be replicated.

| 12 | 4:4:4 YCbCr | 24 | Each 24-bit YCbCr pixel is output in the following order: |
|---|---|---|---|

| **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |
| 2 | Cb7 | Cb6 | Cb5 | Cb4 | Cb3 | Cb2 | Cb1 | Cb0 |
| 3 | Cr7 | Cr6 | Cr5 | Cr4 | Cr3 | Cr2 | Cr1 | Cr0 |

Repeat for the next pixel.

| 13 | 4:0:0 Y | 8 | The Cb and Cr components of the YCbCr pixel are dropped and only the 8-bit Y component is output (this results in a grayscale image): |
|---|---|---|---|

| **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |

Repeat for the next pixel

| 14 | RAWBPA | 8 | Each 8-bit pixel value is output with (almost) no color processing. Only auto exposure, auto white balance and BPA are active: |
|---|---|---|---|

Odd lines (1st, 3rd, 5th, etc.)

| **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |
| 2 | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |

Even lines (2nd, 4th, 6th, etc.)

| **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| 2 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |

**Available output formats (continued)**

| Bits[3:0] | Format | Bits/ Pixel | Description | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 15 | RAW | 8 | Each 8-bit pixel value is output with (almost) no color processing. Only auto exposure is active: | | | | | | | |
| | Odd lines (1st, 3rd, 5th, etc.) | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |
| | | | 2 | R7 | R6 | R5 | R4 | R3 | R2 | R1 | R0 |
| | Even lines (2nd, 4th, 6th, etc.) | | **Byte #** | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| | | | 1 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
| | | | 2 | G7 | G6 | G5 | G4 | G3 | G2 | G1 | G0 |

## The JUST Bit

The JUST bit modifies how the padding with zeros is done for output formats that use padding – RGB 666B, RGB 444B and RGB 444C. The JUST bit is bit 15 in the OUTPUT_STYLE register. Alternatively the JUST bits in the OUTPUT_CTRL_V register and the OUTPUT_CTRL_S register can be used. The OUTPUT_CTRL register can also be used.

# Start of Frame / End of Frame Codes

Whether or not the Start of Frame (SOF) and End of Frame (EOF) codes are included before or after transmitted data is controlled by bits 13 and 14 in the OUTPUT_CTRL_V and OUTPUT_CTRL_S registers. Bit 13 enables the SOF codes and bit 14 enables the EOF codes. The default is for SOF/EOF codes to be off.


**Figure 66** SOF/EOF codes

The start of frame codes for video and still modes are defined in the SOF_CODES register. The EOF_CODES register defines both the good and bad end of frame codes for both still and video.

**Frame codes registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control:<br>Bit 13:SOF codes enable<br>Bit 14:EOF codes enable | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control:<br>Bit 13:SOF codes enable<br>Bit 14:EOF codes enable | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control:<br>Bit 13:SOF codes enable<br>Bit 14:EOF codes enable | 0x9019 | 338 |
| SOF_CODES | 0x0146 | SOF codes for video and still modes | 0xfeff | 341 |
| EOF_CODES | 0x0148 | EOF codes for video and still modes | 0x0100 | 342 |

# CCIR 656 Embedded Synchronization

If embedded synchronization is required, the format of the data needs to be modified. The normal data range is 0 to 255. When embedded synchronization is used, CCIR 656 requires that the data be clipped so that the 0x00 and 0xff data words are reserved. The R_Y_MAX_MIN (address 0x1012, see page 344), the G_CB_MAX_MIN (address 0x1014, see page 345) and the B_CR_MAX_MIN (address 0x1016, see page page 346) registers can be used to clip the data. To meet ITU 601 digital video specifications, which is what the 656 interface normally transmits, set the clip limit registers to the following values.

**CCIR data clipping registers**

| Register name | Address | Value for CCIR embedded synchronization | Default | Page |
|---|---|---|---|---|
| R_Y_MAX_MIN | 0x1012 | Set to 0xeb10:<br>Maximum: 235 = 0xeb<br>Minimum: 16 = 0x10 | 0xff00 | 344 |
| G_CB_MAX_MIN | 0x1014 | Set to 0xf010:<br>Maximum: 240 = 0xf0<br>Minimum: 16 = 0x10 | 0xff00 | 345 |
| B_CR_MAX_MIN | 0x1016 | Set to 0xf010:<br>Maximum: 240 = 0xf0<br>Minimum: 16 = 0x10 | 0xff00 | 346 |

## Enabling Embedded Synchronization Codes

Embedded synchronization codes are controlled by the ESYNC bit in the following registers:

**ESYNC control**

| Register name | Address | Control Bit | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | ESYNC (bit 11), 0 = disabled, 1 = enabled | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | ESYNC (bit 11), 0 = disabled, 1 = enabled | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | ESYNC (bit 11), 0 = disabled, 1 = enabled | 0x9019 | 338 |

Use the OUTPUT_CTRL_V and OUTPUT_CTRL_S to set the embedded synchronization mode for video and still modes, respectively. To change the esync mode without changing picture modes, write to the OUTPUT_CTRL register.

## Embedded HSYNC and VSYNC

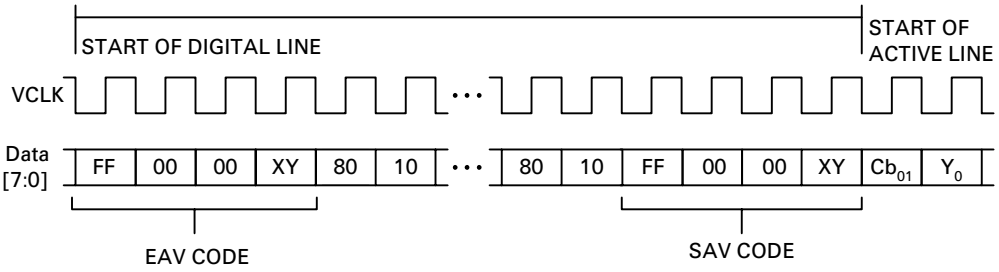



**Figure 67** Embedded VSYNC and HSYNC

## EAV and SAV "XY" Codes

**EAV and SAV codes**

| | | "X" | | | | "Y" | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |
| **Normal values for ADCM-2700** | **Hex** | **"1"** | **F** | **V** | **H** | **P3** | **P2** | **P1** | **P0** |
| SAV | C7 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| EAV | DA | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| SAV, vertical blank | EC | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| EAV, vertical blank | F1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |

F Video field in interlaced systems. 0 for odd field, 1 for even field.
Since the camera module is a progressive system, F is always 1, representing an even field.

V Vertical blanking – 1 during vertical blanking, 0 during active line

H 0 for SAV, 1 for EAV

P0 $F \oplus V \oplus H$ $\oplus$ is the exclusive OR function

P1 $F \oplus V$

P2 $F \oplus H$

P3 $V \oplus H$

## Error Correction

If the data received does not match the valid values on the previous page, use the following table to see if error correction is possible. The received data for F, V and H bits is at the top of the table and the received data protection bits are on the left. At the intersection of the two bits, it is the error corrected version of the F, V and H bits. If the intersection is empty, the error was uncorrectable.

**Error correction table**

| | | | | Received F, V, H Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **P3** | **P2** | **P1** | **P0** | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| 0 | 0 | 0 | 0 | 000 | 000 | 000 | | 000 | | | 111 |
| 0 | 0 | 0 | 1 | 000 | | | 111 | | 111 | 111 | 111 |
| 0 | 0 | 1 | 0 | 000 | | | 011 | | 101 | | |
| 0 | 0 | 1 | 1 | | | 010 | | 100 | | | 111 |
| 0 | 1 | 0 | 0 | 000 | | | 011 | | | 110 | |
| 0 | 1 | 0 | 1 | | 001 | | | 100 | | | 111 |
| 0 | 1 | 1 | 0 | | 011 | 011 | 011 | 100 | | | 011 |
| 0 | 1 | 1 | 1 | 100 | | | 011 | 100 | 100 | 100 | |
| 1 | 0 | 0 | 0 | 000 | | | | | 101 | 110 | |
| 1 | 0 | 0 | 1 | | 001 | 010 | | | | | 111 |
| 1 | 0 | 1 | 0 | | 101 | 010 | | 101 | 101 | | 101 |
| 1 | 0 | 1 | 1 | 010 | | 010 | 010 | | 101 | 010 | |
| 1 | 1 | 0 | 0 | | 001 | 110 | | 110 | | 110 | 110 |
| 1 | 1 | 0 | 1 | 001 | 001 | | 001 | | 001 | 110 | |
| 1 | 1 | 1 | 0 | | | | 011 | | 101 | 110 | |
| 1 | 1 | 1 | 1 | | 001 | 010 | | 100 | | | |

# 6 Synchronous Serial Output Protocols





The ADCM-2700 camera module has the option of using a synchronous serial port to output the data instead of the default parallel port. Since the data is output only 1-bit per clock instead of one byte, see "Notes on Clocks when using Serial Output" on page 129 for additional information on how to set up the clocks in the camera module to ensure complete, untruncated pictures for all possible output formats. The default values for the clocks must be changed if the data is to be output in synchronous serial mode.

The default protocol is for the data to change on falling edges of the clock and be valid on the rising edges.

The following pages contain references to registers used to control the ADCM-2700. When a feature is discussed, there are usually three registers: a video register, a still register and the working register. The video and still registers are used by the camera module and the contents of the currently enabled mode register will be written to the working mode register.

**CAUTION** Do not write directly to the working register. Write new values to the video or still register, stop the camera (0x0000 to register 0x0002) and then start the camera (0x0001 to register 0x0002).

# Serial Output Protocol

The serial mode outputs the same formatted data as the parallel mode, but it is serialized onto the Data[0] pin. The serial mode is flexible and there are many options.

The terminology is slightly different than the parallel mode. The VCLK line is now the data ready line (DRDY), which is a synchronous clock. HSYNC is end of line (EOL) and VSYNC is end of frame (EOF).

**Serial output lines**

| Serial Line | Description | Equivalent Parallel Line |
|---|---|---|
| DRDY | Data ready | VCLK |
| D[0] | Serial data output | Parallel data bit 0 |
| D[1-7] | Unused | Parallel data, bits 1 through 7 |
| EOF | End of frame | VSYNC – vertical synchronization |
| EOL | End of line | HSYNC – horizontal synchronization |

## Enabling Serial Output

The type of output is controlled by bit 0 of the SERIAL_CTRL register. The default output type is parallel.

**Serial output control register**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| SERIAL_CTRL | 0x108e | Serial control<br>Serial data protocol, data order, synchronization controls | 0x0000 | 385 |

Set bit 0 to “1” to enable the synchronous serial port.

## Serial Data Byte Format

The default output format is eight bits of data with the MSB first, followed by an EOL bit and an EOF bit. This is followed by a framing bit, which is always low. All data is valid on the rising edge of DRDY.

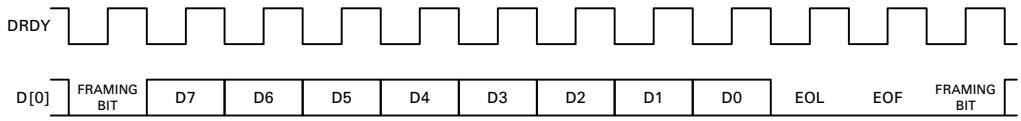



**Figure 68** Default synchronous serial byte format

## Operating Parameters Registers

The following registers control the output protocol of the ADCM-2700:

**Operating parameters registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control:<br>Bit 8: Free running VCLK<br>Bit 9: EOF/EOL protocol<br>Bit 11: Embedded synchronization | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control:<br>Bit 8: Free running VCLK<br>Bit 9: EOF/EOL protocol<br>Bit 11: Embedded synchronization | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control:<br>Bit 8: Free running VCLK<br>Bit 9: EOF/EOL protocol<br>Bit 11: Embedded synchronization | 0x9019 | 338 |
| SERIAL_CTRL | 0x108e | Synchronous serial control:<br>Bit 0: Serial protocol<br>Bit 1: Data order<br>Bit 2: EOL bit<br>Bit 3: EOF bit<br>Bit 4: EOL/EOF order<br>Bit 5: EOL/EOF, data order<br>Bits 6, 7: EOL mode<br>Bits 8, 9: EOF mode | 0x0000 | 385 |

# Serial Data Byte Format Options

## DRDY Polarity

The polarity of the DRDY line is controlled by the P_VCLK bit in the OUTPUT_CTRL register.

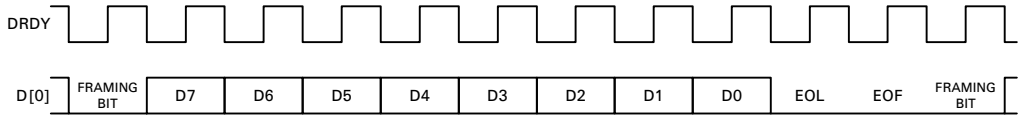



**Figure 69** Normal DRDY polarity (data valid on rising edge), P_VCLK = 0

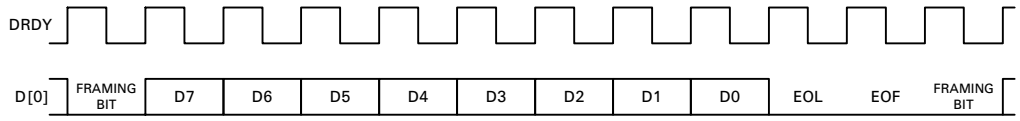



**Figure 70** Inverted DRDY polarity (data valid on falling edge), P_VCLK = 1

## Bit Order

The output order of the data byte can be reversed. This is controlled by the SDO bit (bit 1) in the SERIAL_CTRL register.

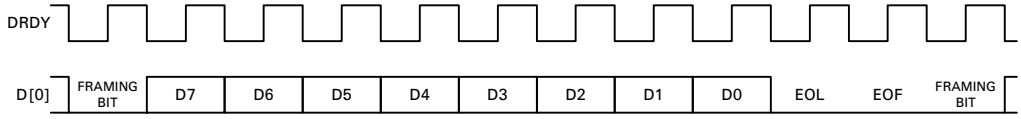



**Figure 71** Serial output data, MSB first, SDO = 0

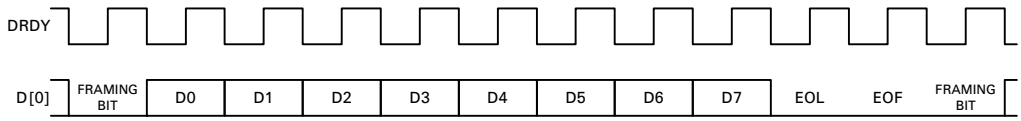



**Figure 72** Serial output data, LSB first, SDO = 1

## DRDY Clocking

The DRDY line is normally free running, but can be programmed to change state only when data is available. FRVCLK (bit 8) in the OUTPUT_CTRL register controls this.

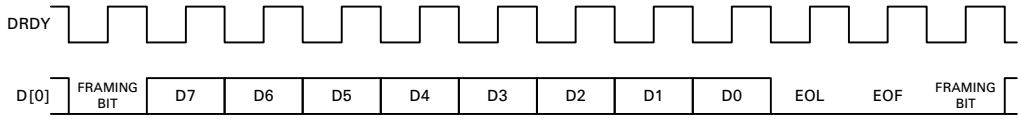



**Figure 73** Free running DRDY, FRVCLK = 0

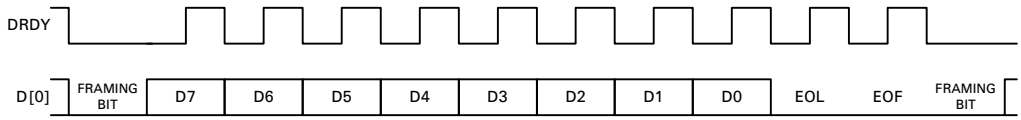



**Figure 74** Gated DRDY, FRVCLK = 1

One side effect of this mode is that the framing bit is not clocked out and if the EOF bit is high, the D[0] line will be high during the unclocked framing bit (refer to the following figure).

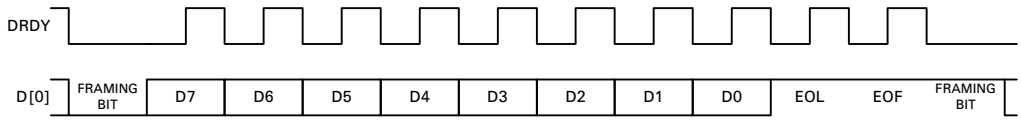



**Figure 75** Gated DRDY, end of frame, FRVCLK = 1

## EOL / EOF Bit Position

The position of the EOL and EOF bits can be either before or after the data (default). The position is controlled by DO (bit 5) in the SERIAL_CTRL register.

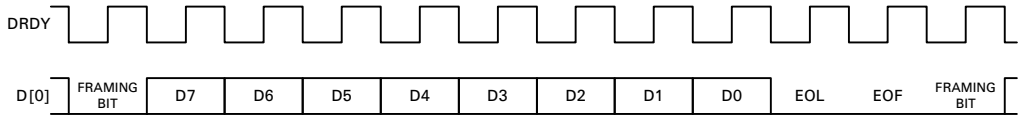



**Figure 76** EOL/EOF bits after data byte, DO = 0

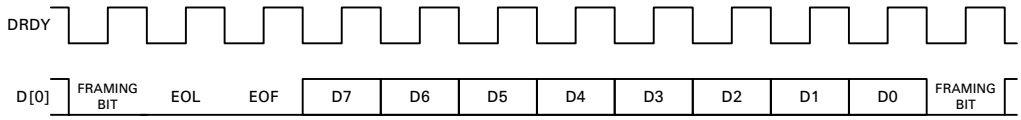



**Figure 77** EOL/EOF bits before data byte, DO = 1

## EOL / EOF Bit Order

The order of the EOL and EOF bits can be defined using the EOL/F_O bit in SERIAL_CTRL (bit 4).

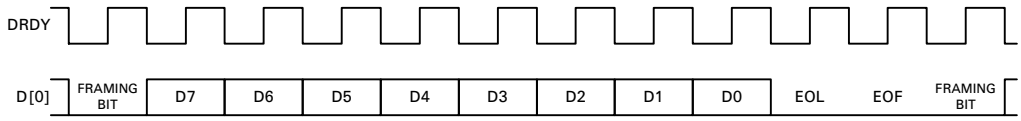



**Figure 78** EOL bit before EOF bit, EOL/F = 0

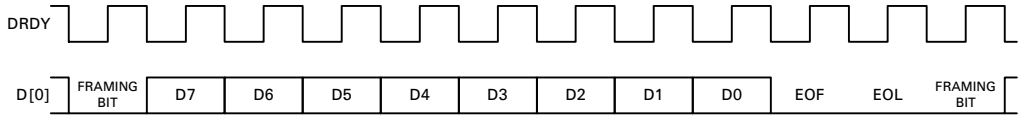



**Figure 79** EOF bit before EOL bit, EOL/F_O = 1

## EOL and EOF Bits Enable

Whether or not to transmit the EOL and EOF bits with the data is controlled by the EOL and EOF bits in SERIAL_CTRL. (bits 2 and 3).

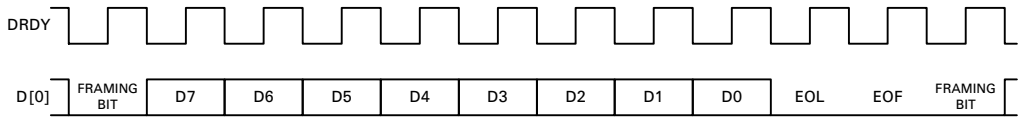



**Figure 80** EOL and EOF enabled, EOL = 0, EOF = 0

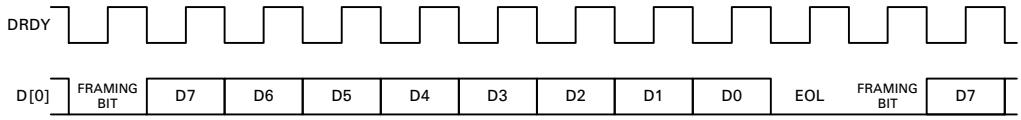



**Figure 81** EOL enabled, EOF disabled: EOL = 0, EOF = 1

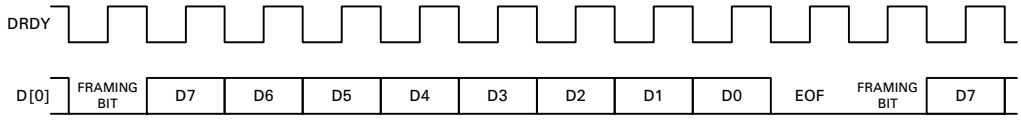



**Figure 82** EOL disabled, EOF enabled: EOL = 1, EOF = 0

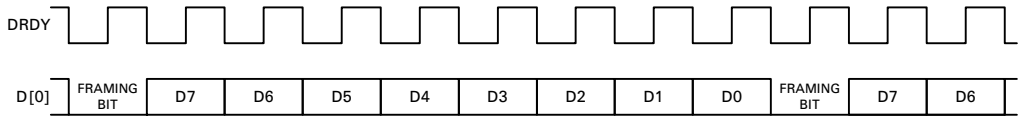



**Figure 83** EOL disabled, EOF disabled: EOL = 1, EOF = 1

# Framing

The data may be framed by inserting EOL and EOF data bits into the data stream (see previous section) and/or using EOL and EOF lines with programmable polarity (see “Inverting Polarities" on page 128). EOL can be programmed to surround words or lines and EOF can be programmed to surround lines or frames.

In the following figures, FB is the framing bit, EL is the embedded EOL bit and EF is the embedded EOF bit. The default is assumed: EOL and EOF bits are enabled and placed after the data and data is MSB first.

## EOL Surrounds Lines, EOF Surrounds Frames


**Figure 84** Beginning of frame

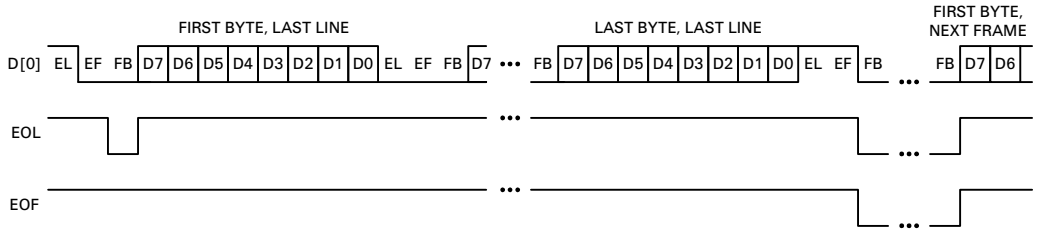



**Figure 85** End of frame

## EOL Surrounds Words, EOF Surrounds Lines

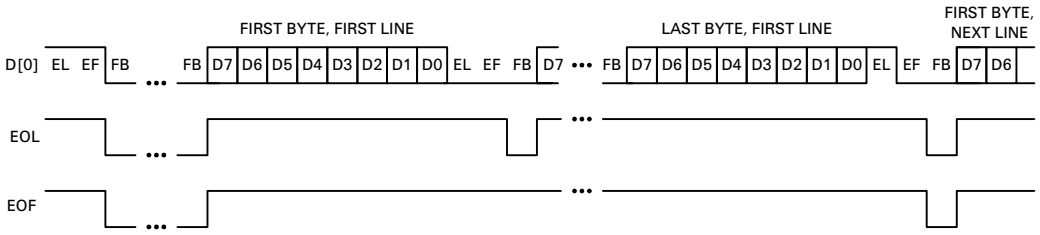



**Figure 86** Beginning of frame

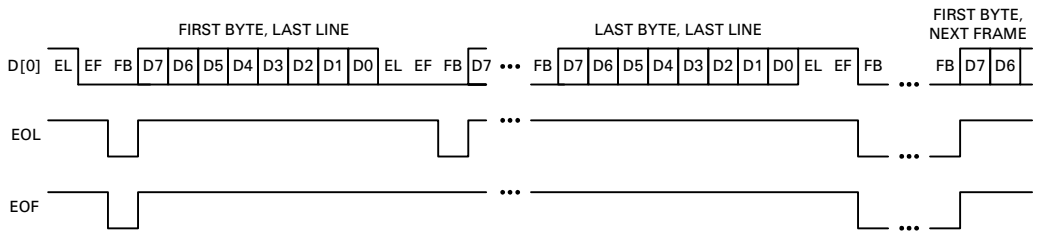



**Figure 87** End of frame

## EOL Surrounds Words, EOF Surrounds Frames

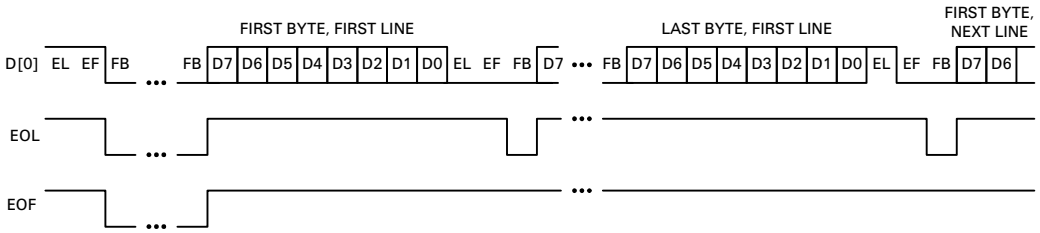



**Figure 88** Beginning of frame

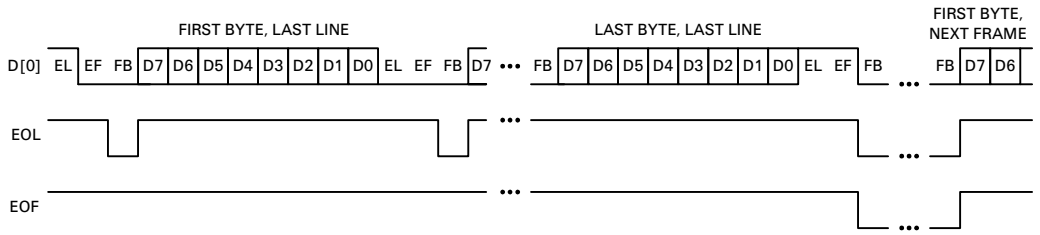



**Figure 89** End of frame

# Inverting Polarities

## VCLK, HSYNC and VSYNC Polarities

The output polarity of the VCLK, HSYNC and VSYNC lines are controlled by the OUTPUT_CTRL_V and OUTPUT_CTRL_S registers. The OUTPUT_CTRL register is written to by the simple control register from the relevant video or still register.

**Operating parameters registers**

| Register name | Address | Description | Default | Page |
|---|---|---|---|---|
| OUTPUT_CTRL_V | 0x0110 | Video mode output control:<br>Bit 5:VCLK polarity<br>Bit 6:VSYNC polarity (inverse EOF)<br>Bit 7:HSYNC polarity (inverse EOL) | 0x9019 | 192 |
| OUTPUT_CTRL_S | 0x0130 | Still mode output control:<br>Bit 5:VCLK polarity<br>Bit 6:VSYNC polarity (inverse EOF)<br>Bit 7:HSYNC polarity (inverse EOL) | 0x8019 | 210 |
| OUTPUT_CTRL | 0x1008 | Current mode output control:<br>Bit 5:VCLK polarity<br>Bit 6:VSYNC polarity (inverse EOF)<br>Bit 7:HSYNC polarity (inverse EOL) | 0x9019 | 338 |

If the VCLK polarity is inverted, then the data changes on the rising edge of VCLK and data is stable and should be read on the falling edge. Similarly, setting the polarity bits for HSYNC or VSYNC will cause those signals to be inverted from their normal levels.

## EOF and EOL Polarities

When in EOL/EOF modes, modes 3 and 4, the EOL and EOF polarities are the inverse of the HSYNC and VSYNC polarities.

# Notes on Clocks when using Serial Output

The simple control register programs the camera module to use the slowest clocks possible for the current output format, assuming a parallel output. It does this to reduce power consumption.

Due to the fact that the serial output is 9 to 11 times slower than the parallel output, the clocks will need to be changed manually. The image pipeline must be able to output data as fast as it receives it from the sensor, there are no internal buffers. If the sensor sends data to the image pipeline faster than it can handle it, the image will be truncated.

## Data Bit Depth

The output data can have different bit depths depending on which output format has been chosen. This has a major impact on the frame rate for the VGA, CIF and QVGA output formats. Refer to the following table for the resulting bit depth for the selected output format.

**Output formats bit depth**

| Bits/Pixel | Format | Bits/Pixel | Format |
|---|---|---|---|
| 24 | 888 RGB | 16 | 4:2:2B CbYCrY |
| 24 | 666B RGB | 16 | 4:2:2C YCrYCb |
| 24 | 444C RGB | 16 | 4:2:2D CrYCbY |
| 24 | 4:4:4 YCbCr | 12 | 444A RGB |
| 18 | 666A RGB | 8 | 332 RGB |
| 16 | 565 RGB | 8 | 4:0:0 Y |
| 16 | 444B RGB | 8 | RAWBPA |
| 16 | 4:2:2A YCbYCr | 8 | RAW |

# Clock Settings: Different Output Formats, Serial Word Lengths, Bit Depths

Since the image pipeline needs to be outputting data at the fastest data rate possible, use the following register settings for each output window size. All settings assume a 13 MHz input MCLK.

**NOTE** If the sensor is programmed to output data faster than the settings shown below, the camera module will truncate the data due to an image pipeline overflow. The output may be correct for the first few lines, but once the image pipeline overflows, all data is lost until the beginning of the next frame.

## VGA Window

Due to the number of pixels, VGA windows cannot be displayed at 15 frames per second. The following numbers optimize the sensor output data rate for maximum frame rate without truncation. The data rate for DRDY is 13 MHz.

**11-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | TBD | TBD | TBD | TBD | TBD |
| SEN_CLK_DIV | 0x0084 | TBD | TBD | TBD | TBD | TBD |
| IP_CLK_DIV | 0x0086 | TBD | TBD | TBD | TBD | TBD |
| PARALLEL_CTRL | 0x100a | TBD | TBD | TBD | TBD | TBD |
| CLK_PIXEL | 0x080e | TBD | TBD | TBD | TBD | TBD |
| HBLANK | 0x0819 | TBD | TBD | TBD | TBD | TBD |
| Frame rate – frames/second | | TBD | TBD | TBD | TBD | TBD |

**10-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | TBD | TBD | TBD | TBD | TBD |
| SEN_CLK_DIV | 0x0084 | TBD | TBD | TBD | TBD | TBD |
| IP_CLK_DIV | 0x0086 | TBD | TBD | TBD | TBD | TBD |
| PARALLEL_CTRL | 0x100a | TBD | TBD | TBD | TBD | TBD |
| CLK_PIXEL | 0x080e | TBD | TBD | TBD | TBD | TBD |
| HBLANK | 0x0819 | TBD | TBD | TBD | TBD | TBD |
| Frame rate – frames/second | | TBD | TBD | TBD | TBD | TBD |

**9-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | TBD | TBD | TBD | TBD | TBD |
| SEN_CLK_DIV | 0x0084 | TBD | TBD | TBD | TBD | TBD |
| IP_CLK_DIV | 0x0086 | TBD | TBD | TBD | TBD | TBD |
| PARALLEL_CTRL | 0x100a | TBD | TBD | TBD | TBD | TBD |
| CLK_PIXEL | 0x080e | TBD | TBD | TBD | TBD | TBD |
| HBLANK | 0x0819 | TBD | TBD | TBD | TBD | TBD |
| Frame rate – frames/second | | TBD | TBD | TBD | TBD | TBD |

## QVGA Window

Due to the number of pixels, QVGA windows cannot be displayed at 15 frames per second. The following numbers optimize the sensor output data rate for maximum frame rate without truncation. The data rate for DRDY is 13 MHz.

**11-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0006 | 0005 | 0004 | 0003 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 15 | 5 | 15 | 15 | 15 |
| Frame rate – frames/second | | 4.65 | 6.02 | 6.75 | 9.01 | 13.50 |

**10-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0005 | 0004 | 0003 | 0003 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 29 | 20 | 4a | 1 | 1 |
| Frame rate – frames/second | | 4.94 | 6.44 | 7.41 | 9.99 | 14.87 |

**9-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0005 | 0004 | 0003 | 0002 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 11 | 3 | 30 | 49 | 0 |
| Frame rate – frames/second | | 5.52 | 7.34 | 7.99 | 10.97 | 14.96 |

## QQVGA, QQQVGA Windows – Using the Sizer

The following register settings are for QQVGA and QQQVGA windows. The data through put with a 13 MHz clock allows 15 frames per second. DRDY is 13 MHz.

| | | All Bit Depths | |
|---|---|---|---|
| **Register name** | **Address** | **QQVGA** | **QQQVGA** |
| CLK_DIV | 0x0080 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0001 | 0001 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 03 | 03 |
| HBLANK | 0x0819 | 4c | 4c |
| Frame rate – frames/second | | 14.96 | 14.96 |

## CIF Window

Due to the number of pixels, CIF windows cannot be displayed at 15 frames per second. The following numbers optimize the sensor output data rate for maximum frame rate without truncation. The data rate for DRDY is 13 MHz.

**11-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0006 | 0005 | 0004 | 0003 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 31 | 1a | 2b | 2b | 2b |
| Frame rate – frames/second | | 3.78 | 5.00 | 5.76 | 7.52 | 11.2 |

**10-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0006 | 0005 | 0004 | 0003 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 1b | 05 | 15 | 15 | 15 |
| Frame rate – frames/second | | 4.13 | 5.49 | 6.21 | 8.26 | 12.44 |

**9-bit serial word registers**

| | | Bit Depth | | | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **24** | **18** | **16** | **12** | **8** |
| CLK_DIV | 0x0080 | 0000 | 0000 | 0000 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0005 | 0004 | 0004 | 0003 | 0002 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 | 0000 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 | 0100 | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 02 | 02 | 02 | 02 | 02 |
| HBLANK | 0x0819 | 2d | 1e | 41 | 0 | 0 |
| Frame rate – frames/second | | 4.61 | 5.99 | 6.89 | 9.09 | 13.89 |

## QCIF, QQCIF Windows – Using the Sizer

The following register settings are for QCIF and QQCIF windows. The data though put with a 13 MHz clock allows 15 frames per second. DRDY is 13 MHz.

| | | All Bit Depths | |
|---|---|---|---|
| **Register name** | **Address** | **QCIF** | **QQCIF** |
| CLK_DIV | 0x0080 | 0000 | 0000 |
| SEN_CLK_DIV | 0x0084 | 0001 | 0001 |
| IP_CLK_DIV | 0x0086 | 0000 | 0000 |
| PARALLEL_CTRL | 0x100a | 0100 | 0100 |
| CLK_PIXEL | 0x080e | 03 | 03 |
| HBLANK | 0x0819 | 36 | 36 |
| Frame rate – frames/second | | 15.0 | 15.0 |

# 7 Simple Control Registers

# Overview

The simple control registers provide an easy to use method for configuring and operating the camera module. Values set into these registers are read by the camera control processor and are used to derive detailed settings for the sensor and image pipeline. Common camera operations can be achieved using only these registers.

In cases where special behavior is needed, the host system can change the detailed settings after they have been created by the control processor.

**Simple control register index**

## ID – Chip ID (Read Only)

Address: 0x0000 Block: 0x00 Offset: 0x00

Access: Read only Reset value: 0x0060 Type: One 3-bit integer, one 5-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TYPE | | | | | REV | | |

**Chip ID**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | REV | 000 | Revision of the IC – first revision |
| 3 - 7 | TYPE | 01100 | Type of IC:<br>0x0c:ADCM-2700 IC |
| 8 - 15 | Reserved | 00000000 | Reserved |

## CONTROL – Camera Control

Address: 0x0002 Block: 0x00 Offset: 0x02

Access: Read / Write Reset value: 0x0001 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CLRSTAT | 0 | 0 | 0 | IS_RST | CONFIG | SNAP | RUN |

**Camera control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | RUN | 1 | Run in viewfinder (video) mode:<br>0: Still mode<br>**1: Video mode** |
| 1 | SNAP | 0 | Take a still image – if the camera is not in the RUN mode with a stable exposure, it will enter the RUN mode with outputs disabled until the exposure is stable and then snap a picture:<br>**0: Disabled**<br>1: Take a still picture |
| 2 | CONFIG | 0 | Configuration – the CONFIG bit causes the camera to use the values of the Control register set to update the detailed hardware registers. After that operation, this bit will be cleared. If the CONFIG and RUN bits are set simultaneously, then the camera will start running. If the CONFIG bit is set but the RUN bit is not, then the host system can alter the detailed register settings before setting the RUN bit:<br>**0: Normal**<br>1: Update the hardware registers |
| 3 | IS_RST | 0 | Image system reset – setting this bit causes the imaging system (sensor, image pipeline and operating parameters) to be reset to their default state. If the IS_RST and CONFIG bits are set simultaneously, the reset will occur before the configuration operation. The IS_RST bit is cleared after the reset has occurred:<br>**0: Normal**<br>1: Reset image system |
| 4 - 6 | Reserved | 000 | Reserved |

**Camera control (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 7 | CLRSTAT | 0 | Clear the status error and code – self resets to zero:<br>**0: No operation**<br>1: Clear status error and code values |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE**

If the RUN, CONFIG and SNAP bits are all set to zero, the camera will enter a SLEEP state to save power. All register values will be retained.

The value of the CONTROL register immediately after a reset is 0x0005, but will change to 0x0001 as soon as initialization is complete.

## STATUS – Camera Status

Address: 0x0004 Block: 0x00 Offset: 0x04

Access: Read only Reset value: 0x0004 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | CF_ERR | IRQ_ERR | WDT_TO | IP_TO | SEN_TO |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ERROR | 0 | 0 | 0 | 0 | CONFIG | SNAP | RUN |

**Camera status**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | RUN | 1 | Run mode (video):<br>0: Stopped or in snapshot mode<br>**1: Running in viewfinder mode** |
| 1 | SNAP | 0 | Snap mode (still):<br>**0: Snapshot mode off, or in viewfinder mode**<br>1: Running in snapshot mode |
| 2 | CONFIG | 0 | Configuration mode:<br>**0: Normal**<br>1: Camera is configuring |
| 3 - 6 | Reserved | 0000 | Reserved |
| 7 | ERROR | 0 | Camera error:<br>**0: No error**<br>1: Error during configuration (see bits 8, 9, 10, 11 and 12) |
| 8 | SEN_TO | 0 | Sensor timeout:<br>**0: No error**<br>1: Sensor did not stop in time |
| 9 | IP_TO | 0 | Image pipeline timeout:<br>**0: No error**<br>1: Image pipeline did not start in time |
| 10 | WDT_TO | 0 | Watchdog timer timeout:<br>**0: No error**<br>1: Watchdog timer expired |
| 11 | IRQ_ERR | 0 | IRQ error:<br>**0: No error**<br>1: IRQ interrupt occurred |

**Camera status (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 12 | CF_ERR | 0 | SCL clock frequency error:<br>**0: No error**<br>1: Serial control clock is out of range |
| 13 - 15 | Reserved | 000 | Reserved |

**NOTE** The value of the status register immediately after a reset will be 0x0004, but will change to 0x0001 as soon as initialization is complete.

## CLK_FREQ – Input Clock Frequency

Address: 0x0006 Block: 0x00 Offset: 0x06

Access: Read / Write Reset value: 0x32c8 Type: 16-bit integer

Decimal: 13,000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| CLK_FREQ | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CLK_FREQ | | | | | | | |

**Input clock frequency**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | CLK_FREQ | 00110010 11001000 | Input clock frequency in kHz – default value is 13 MHz, 13,000 kHz |

## SIZE – Image Size and Orientation

Address: 0x0008 Block: 0x00 Offset: 0x08

Access: Read / Write Reset value: 0x0605 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SSUB | SPREF | SFLIP_UD | SFLIP_LR | 0 | SSIZE | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VSUB | VPREF | VFLIP_UD | VFLIP_LR | 0 | VSIZE | | |

**Image size and orientation**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | VSIZE | 01 | Size for viewfinder mode:<br>000: QQCIF – landscape 88 x 72<br>001: QCIF – landscape 176 x 144<br>010: CIF – landscape 352 x 288<br>011: QQQVGA – landscape 80 x 60<br>100: QQVGA – landscape 160 x 120<br>**101: QVGA – landscape 320 x 240**<br>110: VGA – landscape 640 x 480<br>111: Use registers 0x0018 through 0x001e to determine window size |
| 3 | Reserved | 0 | Reserved – hard wired to 0 |
| 4 | VFLIP_LR | 0 | Controls left – right flipping for viewfinder mode (horizontal flip):<br>**0: Not flipped**<br>1: Flipped |
| 5 | VFLIP_UD | 0 | Controls up – down flipping for viewfinder mode (vertical flip):<br>**0: Not flipped**<br>1: Flipped |
| 6 | VPREF | 0 | Power / image quality preference in viewfinder mode:<br>**0: Prefer image quality**<br>1: Prefer lower power |
| 7 | VSUB | 0 | Allow sensor subsampling in video mode – only active if VPREF = 1:<br>**0: Disallow sensor subsampling**<br>1: Allow sensor subsampling |

**Image size and orientation (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 - 10 | SSIZE | 10 | Size for still mode:<br>000: QQCIF – landscape 88 x 72<br>001: QCIF – landscape 176 x 144<br>010: CIF – landscape 352 x 288<br>011: QQQVGA – landscape 80 x 60<br>100: QQVGA – landscape 160 x 120<br>101: QVGA – landscape 320 x 240<br>**110: VGA – landscape 640 x 480**<br>111: Use registers 0x0020 through 0x0026 to determine window size |
| 11 | Reserved | 00 | Reserved – hard wired to 0 |
| 12 | SFLIP_LR | 0 | Controls left – right flipping for still mode (horizontal flip):<br>**0: Not flipped**<br>1: Flipped |
| 13 | SFLIP_UD | 0 | Controls up – down flipping for still mode (vertical flip):<br>**0: Not flipped**<br>1: Flipped |
| 14 | SPREF | 0 | Power / image quality preference in still mode:<br>**0: Prefer image quality**<br>1: Prefer lower power |
| 15 | SSUB | 0 | Allow sensor subsampling in still mode – only active if SPREF = 1:<br>**0: Disallow sensor subsampling**<br>1: Allow sensor subsampling |

**NOTE** Output size is achieved by using the sizer for QVGA, QQVGA and QQQVGA. For CIF, QCIF and QQCIF a combination of windowing to change the aspect ratio and the sizer to change the size are used.

## OUTPUT_FORMAT – Output Format

Address: 0x000a Block: 0x00 Offset: 0x0a

Access: Read / Write Reset value: 0x0909 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | SOUT | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | VDIS | VOUT | | | |

**Output format**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | VOUT | 1001 | Output format for viewfinder mode:<br>0000: RGB 888 in 3 bytes<br>0001: RGB 666 A, 4 pixels in 9 bytes<br>0010: RGB 666 B, 1 pixel in 3 bytes, left or right justified<br>0011: RGB 565, 1 pixel in 2 bytes<br>0100: RGB 444 A, 2 pixels in 3 bytes, RG BR GB<br>0101: RGB 444 B, 1 pixel in 2 bytes, RG B0 or 0R GB<br>0110: RGB 444 C, 1 pixels in 3 bytes, R0 G0 B0 or 0R 0G 0B<br>0111: RGB 332, 1 pixel in 1 byte<br>1000: YCbCr (YUV) 4:2:2 A – Y1, U12, Y2, V12 order<br>**1001: YCbCr (YUV) 4:2:2 B – U12, Y1, V12, Y2 order**<br>1010: YCbCr (YUV) 4:2:2 C – Y1, V12, Y2, U12 order<br>1011: YCbCr (YUV) 4:2:2 D – V12, Y1, U12, Y2 order<br>1100: YCbCr (YUV) 4:4:4<br>1101: Grayscale – Y 4:0:0<br>1110: Raw sensor data (8 bits per pixel)<br>1111: Raw sensor data (8 bits per pixel) – no BPA correction |
| 4 | VDIS | 0 | Disable camera output – camera will run but no output will be generated; useful to acquire exposure and other settings:<br>**0: Camera output enabled**<br>1: Camera output disabled |
| 5 - 7 | Reserved | 000 | Reserved – hard wired to 0 |

**Output format (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 - 11 | SOUT | 1001 | Output format for still mode:<br>0000: RGB 888 in 3 bytes<br>0001: RGB 666 A, 4 pixels in 9 bytes<br>0010: RGB 666 B, 1 pixel in 3 bytes, left or right justified<br>0011: RGB 565, 1 pixel in 2 bytes<br>0100: RGB 444 A, 2 pixels in 3 bytes, RG BR GB<br>0101: RGB 444 B, 1 pixel in 2 bytes, RG B0 or 0R GB<br>0110: RGB 444 C, 1 pixels in 3 bytes, R0 G0 B0 or 0R 0G 0B<br>0111: RGB 332, 1 pixel in 1 byte<br>1000: YCbCr (YUV) 4:2:2 A – Y1, U12, Y2, V12 order<br>**1001: YCbCr (YUV) 4:2:2 B – U12, Y1, V12, Y2 order**<br>1010: YCbCr (YUV) 4:2:2 C – Y1, V12, Y2, U12 order<br>1011: YCbCr (YUV) 4:2:2 D – V12, Y1, U12, Y2 order<br>1100: YCbCr (YUV) 4:4:4<br>1101: Grayscale – Y 4:0:0<br>1110: Raw sensor data (8 bits per pixel)<br>1111: Raw sensor data (8 bits per pixel) – no BPA correction |
| 12 - 15 | Reserved | 0000 | Reserved – hard wired to 0 |

**NOTE** For additional information on each output format, see Chapter 5, “Data Output Formats,” starting on page 105.

## EXPOSURE – Exposure Time

Address: 0x000c Block: 0x00 Offset: 0x0c

Access: Read / Write Reset value: 0x03e8 Type: 16-bit integer

Decimal: 10 ms

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| ETIME | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ETIME | | | | | | | |

**Exposure time**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | ETIME | 00000011 11101000 | Exposure time in 10 microsecond units – (maximum time = 0.65 seconds), default = 10 ms |

**NOTE** This register contains the current exposure value when the camera module is running. If EMOD is set in EXP_ADJ, ETIM will be the current exposure used by the camera module.

## EXP_ADJ – Exposure Adjustment

Address: 0x000e Block: 0x00 Offset: 0x0e

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| EMOD | 0 | 0 | 0 | 0 | 0 | EXPZONE | ECMPSGN |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | ECOMP | | | |

**Exposure adjustment**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | ECOMP | 0000 | Exposure compensation – units are 0.25 f-stops and valid values are 0 to 3 f-stops:<br>**0000: 0** 1000: 2.00 f-stop<br>0001: 0.25 f-stop 1001: 2.25 f-stop<br>0010: 0.50 f-stop 1010: 2.50 f-stop<br>0011: 0.75 f-stop 1011: 2.75 f-stop<br>0100: 1.00 f-stop 1100: 3.00 f-stop<br>0101: 1.25 f-stop 1101: reserved<br>0110: 1.50 f-stop 1101: reserved<br>0111: 1.75 f stop 1111: reserved |
| 4 - 7 | Reserved | 0000 | Reserved – hard wired to 0 |
| 8 | ECMPSGN | 0 | Exposure compensation sign:<br>**0: Plus (expose more)**<br>1: Minus (expose less) |
| 9 | EXPZONE | 0 | Exposure zone:<br>**0: Full frame**<br>1: Center zone |
| 10 - 14 | Reserved | 00000 | Reserved – hard wired to 0 |
| 15 | EMOD | 0 | Exposure mode:<br>**0: Auto: EXPOSURE reads actual exposure time**<br>1: Manual: Use EXPOSURE value for exposure time |

**NOTE** When changing the exposure zone from center to full frame, also write a value of 0x0000 to register 0x1074 (STAT_MODE_CTRL) after writing to EXP_ADJ to fix a firmware bug.

## ILLUM – Illumination

Address: 0x0010 Block: 0x00 Offset: 0x10

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | FLICKER | | I_TYPE | | AWB |

**Illumination**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | AWB | 0 | Automatic white balance mode:<br>**0: AWB on**<br>1: AWB off |
| 1 - 2 | I_TYPE | 00 | Illumination type:<br>**00: Default**<br>01: Daylight (D65)<br>10: Fluorescent<br>11: Tungsten |
| 3 - 4 | FLICKER | 00 | Flicker mode:<br>**00: Auto**<br>01: 50 Hz<br>10: 60 Hz<br>11: Reserved |
| 5 - 15 | Reserved | 00000000000 | Reserved – hard wired to 0 |

**NOTE** When using fixed illuminants, turn auto white balance off.

## FRAME_RATE – Frame Rate

Address: 0x0012 Block: 0x00 Offset: 0x12

Access: Read / Write Reset value: 0x0096 Type: 9-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | FRATE |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FRATE | | | | | | | |

**Frame rate**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | FRATE | 0<br>10010110 | Frame rate in 1/10 frames per second – default is 150 = 15.0 frames per second |
| 9 - 15 | Reserved | 0000000 | Reserved – hard wired to 0 |

**NOTE**

The actual frame rate may not exactly match this value. Many factors limit the exact frame rate. The control processor will select the closest possible value to the value specified in the FRAME_RATE register.

## RESERVED – Reserved

Address: 0x0014 Block: 0x00 Offset: 0x14

Access: Read / Write Reset value: 0x0000 Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to these registers |

## A_FRAME_RATE – Actual Frame Rate (Read only)

Address: 0x0016 Block: 0x00 Offset: 0x12

Access: Read only Reset value: 0x0096 Type: 9-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | AFRATE |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AFRATE | | | | | | | |

**Actual frame rate**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | AFRATE | 0<br>10010110 | Frame rate in 1/10 frames per second – default is 150 = 15.0 frames per second |
| 9 - 15 | Reserved | 0000000 | Reserved – hard wired to 0 |

**NOTE** This register is the actual frame rate achievable with the current settings of PLL, clock divisors, window size and subsampling.

## SENSOR_WID_V – Sensor Window Width, Video Mode

Address: 0x0018 Block: 0x00 Offset: 0x18

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SEN_WID_V | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SEN_WID_V | | | | | | | |

**Sensor window width, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | SEN_WID_V | 00000000 | Sensor window width, video mode: specifies the width of the sensor window for video (viewfinder) mode. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [2:0] of the SIZE register (address 0x0008) are equal to 111.

## SENSOR_HGT_V – Sensor Window Height, Video Mode

Address: 0x001a Block: 0x00 Offset: 0x1a

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SEN_HGT_V | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SEN_HGT_V | | | | | | | |

**Sensor window height, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | SEN_HGT_V | 00000000 | Sensor window height, video mode: specifies the height of the sensor window for video (viewfinder) mode. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [2:0] of the SIZE register (address 0x0008) are equal to 111.

## OUTPUT_WID_V – Output Window Width, Video Mode

Address: 0x001c Block: 0x00 Offset: 0x1c

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| OUT_WID_V | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_WID_V | | | | | | | |

**Output window width, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | OUT_WID_V | 00000000 | Output window width, video mode: specifies the width of the output window for video (viewfinder) mode. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [2:0] of the SIZE register (address 0x0008) are equal to 111.

## OUTPUT_HGT_V – Output Window Height, Video Mode

Address: 0x001e Block: 0x00 Offset: 0x1e

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| OUT_HGT_V | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_HGT_V | | | | | | | |

**Output window height, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | OUT_HGT_V | 00000000 | Output window height, video mode: specifies the height of the output window for video (viewfinder) mode. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [2:0] of the SIZE register (address 0x0008) are equal to 111.

## SENSOR_WID_S – Sensor Window Width, Still Mode

Address: 0x0020 Block: 0x00 Offset: 0x20

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SEN_WID_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SEN_WID_S | | | | | | | |

**Sensor window width, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | SEN_WID_S | 00000000 | Sensor window width, still mode: specifies the width of the sensor window for still operation. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [10:8] of the SIZE register (address 0x0008) are equal to 111.

## SENSOR_HGT_S – Sensor Window Height, Still Mode

Address: 0x0022 Block: 0x00 Offset: 0x22

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SEN_HGT_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SEN_HGT_S | | | | | | | |

**Sensor window height, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | SEN_HGT_S | 00000000 | Sensor window height, still mode: specifies the height of the sensor window for still operation. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [10:8] of the SIZE register (address 0x0008) are equal to 111.

## OUTPUT_WID_S – Output Window Width, Still Mode

Address: 0x0024 Block: 0x00 Offset: 0x24

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| OUT_WID_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_WID_S | | | | | | | |

**Output window width, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | OUT_WID_S | 00000000 | Output window width, still mode: specifies the width of the output window for still operation. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [10:8] of the SIZE register (address 0x0008) are equal to 111.

## OUTPUT_HGT_S – Output Window Height, Still Mode

Address: 0x0026 Block: 0x00 Offset: 0x26

Access: Read / Write Reset value: 0x0000 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| OUT_HGT_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_HGT_S | | | | | | | |

**Output window height, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 16 | OUT_HGT_S | 00000000 | Output window height, still mode: specifies the height of the output window for still operation. The bottom three bits will be truncated, resulting in an even multiple of 8. |

**NOTE** This register is only active when bits [10:8] of the SIZE register (address 0x0008) are equal to 111.

## RESERVED – Reserved

Address: 0x0028 ⋮ 0x00fe  Block: 0x01  Offset: 0x28 ⋮ 0x7e

Access: Read / Write  Reset Value:–  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved registers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | – | Reserved – DO NOT write any data to these registers |

# 8 Expert Hardware Registers



## Overview

The following registers are automatically programmed by the control block when operating in automatic mode. To make changes to any of these registers, do so after setting the CONFIG bit in the CONTROL register.

**Expert control register index**

| Mnemonic | Page | Mnemonic | Page | Mnemonic | Page |
|---|---|---|---|---|---|
| CLK_DIV | 166 | PLL_CTRL | 174 | SER_ADDR | 171 |
| CTL_CLK_DIV | 167 | PLL_DIV_H | 175 | SER_PARM | 172 |
| IP_CLK_DIV | 169 | PLL_DIV_L | 176 | TST_MODE | 170 |
| OUT_CTRL | 173 | SEN_CLK_DIV | 168 | | |

## I_CLK_DIV – Initial Clock Divider

Address: 0x0080 Block: 0x01 Offset: 0x00

Access: Read / Write Reset value: 0x0001 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | PRESCALE | | |

**Initial clock divider**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | PRESCALE | 01 | Divider value for MCLK prescaler: PRESCALE +1<br>000: /1<br>**001: /2**<br>010: /3<br>011: /4<br>100: /5<br>101: /6<br>110: /7<br>111: /8 |
| 3 - 15 | Reserved | 00000000 00000 | Reserved |

## CTL_CLK_DIV – Clock Dividers Control and Serial Interface

Address: 0x0082 Block: 0x01 Offset: 0x02

Access: Read / Write Reset value: 0x4000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| CPU_RST | WDDIS | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | CTL_CLK | | |

**Clock dividers for control and serial interface**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | CTL_CLK | 000 | Divider value for control and synchronous serial interfaces; value used is n x 2, if n = 0, value is 1<br>**0000: /1**<br>0001: /2<br>0010: /4<br>0011: /6<br>0100: /8<br>0101: /10<br>0110: /12<br>0111: /14 |
| 3 - 13 | Reserved | 000000 00000 | Reserved |
| 14 | WDDIS | 1 | Watchdog disable (debug only):<br>0: Normal watchdog timer operation<br>**1: Disable the processor watchdog timer** |
| 15 | CPU_RST | 0 | Reset the CPU:<br>**0: Normal**<br>1: Reset |

## SEN_CLK_DIV – Sensor Clock Dividers

Address: 0x0084 Block: 0x01 Offset: 0x04

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SEN_RST | SEN_OFF | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | SEN_CLK | | |

**Sensor clock dividers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | SEN_CLK | 000 | Divider value for sensor clock; value used is n x 2, if n = 0, value is 1<br>**000: /1**<br>001: /2<br>010: /4<br>011: /6<br>100: /8<br>101: /10<br>110: /12<br>111: /14 |
| 3 - 13 | Reserved | 000000 00000 | Reserved |
| 14 | SEN_OFF | 0 | Turns off all clocks to sensor:<br>**0: Normal**<br>1: Clocks off |
| 15 | SEN_RST | 0 | Reset sensor:<br>**0: Normal**<br>1: Reset |

## IP_CLK_DIV – Image Pipeline Clock Dividers

Address: 0x0086 Block: 0x01 Offset: 0x06

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| IP_RST | IP_OFF | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | IP_CLK | | |

**Image pipeline clock dividers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | IP_CLK | 000 | Divider value for image pipeline clock; value used is n x 2, if n = 0, value is 1:<br>**000: /1**<br>001: /2<br>010: /4<br>011: /6<br>100: /8<br>101: /10<br>110: /12<br>111: /14 |
| 3 - 13 | Reserved | 000000 00000 | Reserved |
| 14 | IP_OFF | 0 | Turn off all clocks to image pipeline:<br>**0: Normal**<br>1: Clocks off |
| 15 | IP_RST | 0 | Reset image pipeline:<br>**0: Normal**<br>1: Reset |

## TST_MODE – Latched Test Mode (Read only)

Address: 0x0088 Block: 0x01 Offset: 0x08

Access: Read only Reset value: 0x0000 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T_MODE | | | | | | | |

**Latched test mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | T_MODE | 00000000 | Value latched from DATA at RESETn |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register is for debug purposes only and should NOT be used.

## SER_ADDR – Serial Interface Device Address

Address: 0x008a Block: 0x01 Offset: 0x0a

Access: Read / Write Reset value: 0x0053 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DEVADDR | | | | | | | |

**Serial interface device address**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | DEVADDR | 01010011 | Serial interface device address |
| 8 - 15 | Reserved | 00000000 | Reserved |

## SER_PARM – Serial Interface Parameters

Address: 0x008c Block: 0x01 Offset: 0x0c

Access: Read / Write Reset value: 0x0000 Type: Two 4-bit integers

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SETUP | | | | HOLD | | | |

**Serial interface parameters**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | HOLD | 0000 | Hold time (units are control clocks) |
| 4 - 7 | SETUP | 0000 | Setup time (units are control clocks) |
| 8 - 15 | Reserved | 00000000 | Reserved |

## OUT_CTRL – Output Control

Address: 0x008e Block: 0x01 Offset: 0x0e

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TRI_CCIR | 0 | FLAG_DIR | FLAG_IN | FLGSRC | | | |

**Output control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | FLGSRC | 0000 | FLAG source:<br>**0000: Always 0**<br>0001: Always 1<br>0010: Sensor frame signal<br>0011: Sensor IRQ signal<br>0100 through 1111: Unused |
| 4 | FLAG_IN | 0 | FLAG input value |
| 5 | FLAG_DIR | 0 | FLAG direction:<br>**0: Output**<br>1: Input |
| 6 | Reserved | 0 | Reserved |
| 7 | TRI_CCIR | 0 | Tri-state the CCIR lines (DATA, HSYNC, VSYNC, VCLK):<br>**0: Normal CCIR lines**<br>1: Tri-stated CCIR lines |
| 8 - 15 | Reserved | 00000000 | Reserved |

## PLL_CTRL – Phase Lock Loop Control

Address: 0x0090 Block: 0x01 Offset: 0x10

Access: Read / Write Reset value: 0x0024 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PLL_LOCK | PLL_I_DIV | | | PLL_DE | PLL_FA | PLL_SE | PLL_EN |

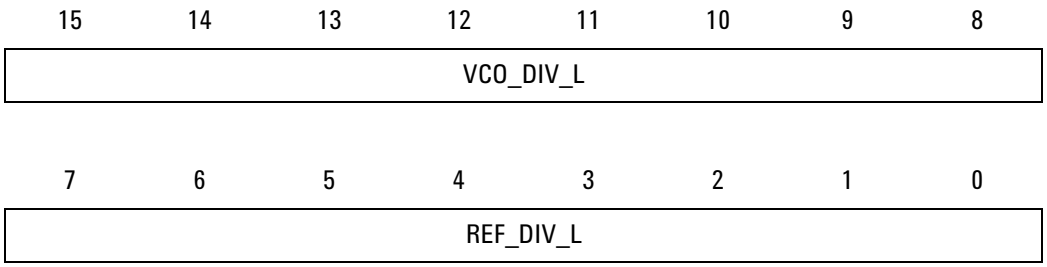
**Phase lock loop control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | PLL_EN | 0 | PLL enable:<br>**0: PLL disabled**<br>1: PLL enabled |
| 1 | PLL_SE | 0 | PLL synchronization enable:<br>**0: Normal operation**<br>1: Synchronization operation |
| 2 | PLL_FA | 1 | PLL fast acquire:<br>0: Normal acquire<br>**1: Fast acquire** |
| 3 | PLL_DE | 0 | PLL dither enable:<br>**0: PLL normal operation**<br>1: PLL dithered operation |
| 4 - 6 | PLL_I_DIV | 010 | PLL initial divide: default = 010 = 2 |
| 7 | PLL_LOCK | 1 | PLL locked (read only):<br>0: Reset PLL<br>**1: PLL normal operation** |
| 8 - 15 | Reserved | 00000000 | Reserved |

## PLL_DIV_L – PLL Large Divisors

Address: 0x0092 Block: 0x01 Offset: 0x12

Access: Read / Write Reset value: 0x1d09 Type: Two 8-bit integers




**PLL large divisors**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | REF_DIV_L | 00001001 | PLL REF divisor, large value |
| 8 - 15 | VCO_DIV_L | 00011101 | PLL VCO divisor, large value |

The PLL output frequency will be:

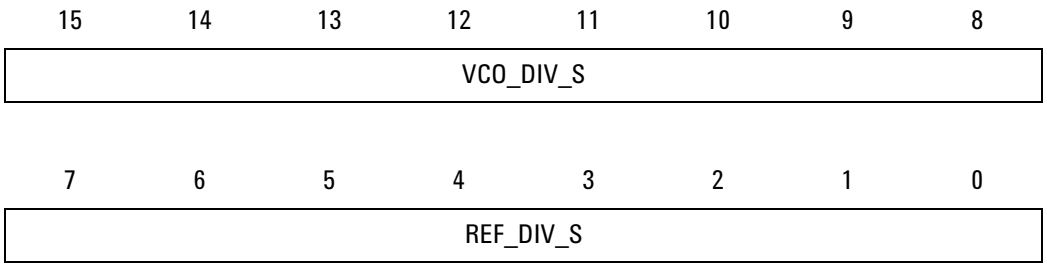
$$\frac{VCO\_DIV\_L + 1}{REF\_DIV\_L + 1} \times MCLK$$

when the PLL is first enabled. After PLL lock has been achieved, the PLL logic will switch to use the small divide values. (less jitter)

## PLL_DIV_S – PLL Small Divisors

Address: 0x0094 Block: 0x01 Offset: 0x14

Access: Read / Write Reset value: 0x0200 Type: Two 8-bit integers




**PLL small divisors**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | REF_DIV_S | 00000000 | PLL REF divisor, small value |
| 8 - 15 | VCO_DIV_S | 00000010 | PLL VCO divisor, small value |

The PLL output frequency will be:

$$\frac{VCO\_DIV\_S + 1}{REF\_DIV\_S + 1} \times MCLK$$

after PLL lock has been achieved.

## RESERVED – Reserved

Address: 0x0096 ⋮ 0x00fe  Block: 0x01  Offset: 0x16 ⋮ 0x7e

Access: Read / Write  Reset Value:–  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | – | Reserved – DO NOT write any data to these registers |

# 9 Expert Camera Controller Registers

## Overview

The registers in this chapter control the operation details of the ADCM-2700. They are written to by the camera controller, with values specified by the simple control registers. These registers can be written to directly.

## Register Description Notes

The register descriptions show the format of the data in each register. Each description starts with the mnemonic name for the register, followed by the full register name. The full 16-bit address is listed, with the block and offset values needed to communicate with the register using the serial control port.

The access, either read/write, read only or mixed is shown, along with the 16-bit reset value (in hex, when the value has a numeric meaning, the decimal equivalent value is shown). The register type is also listed: bit field, real, integer, 2's complement, etc.

The 16 bits of the register are shown, with mnemonics for each bit. Whenever a bit is either unused or reserved, the value for the bit (one or zero) is shown instead of the mnemonic and the block is grayed out.

Each register contains a table with details for each bit. Shown are the bit location in the register, the mnemonic, the reset value of the bits and the description of all of the valid values of the bit. The default value is shown in **bold** type.

## Register index

## Register index (continued)

# Video Registers

## SZR_IN_WID_V – Sizer Input Width, Video Mode

Address: 0x0100 Block: 0x02 Offset: 0x00

Access: Read / Write Reset value: 0x0280 Type: 10-bit integer

Decimal: 640

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | IN_WID_V | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IN_WID_V | | | | | | | |

**Sizer input width – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | IN_WID_V | 10<br>10000000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE**

This register writes to image pipeline register "SZR_IN_W – Sizer Input Width" on page 352, when the module is in video mode.

## SZR_IN_HGT_V – Sizer Input Height, Video Mode

Address: 0x0102 Block: 0x02 Offset: 0x02

Access: Read / Write Reset value: 0x01e0 Type: 10-bit integer

Decimal: 480

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | I_HGT_V | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| I_HGT_V | | | | | | | |

**Sizer input height – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | I_HGT_V | 01 11100000 | Number of rows in one frame (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** This register writes to image pipeline register "SZR_IN_H – Sizer Input Height" on page 353, when the module is in video mode.

## SZR_OUT_WID_V – Sizer Output Width, Video Mode

Address: 0x0104 Block: 0x02 Offset: 0x04

Access: Read / Write Reset value: 0x0140 Type: 10-bit integer

Decimal: 320

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | O_WID_V | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| O_WID_V | | | | | | | |

**Sizer output width – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | O_WID_V | 01 01000000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE**

This register writes to image pipeline register "SZR_OUT_W – Sizer Output Width" on page 354, when the module is in video mode.

## SZR_OUT_HGT_V – Sizer Output Height - Video Mode

Address: 0x0106 Block: 0x02 Offset: 0x06

Access: Read / Write Reset value: 0x00f0 Type: 9-bit integer

Decimal: 240

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | O_HGT_V | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| O_HGT_V | | | | | | | |

**Sizer output height – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | O_HGT_V | 00 11110000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** This register writes to image pipeline register "SZR_OUT_H – Sizer Output Height" on page 355, when the module is in video mode.

## CPP_V – Clocks per Pixel, Video Mode

Address: 0x0108 Block: 0 0x02 Offset: 0x08

Access: Read / Write Reset value: 0x0002 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CPP_V | | | | | | | |

**Clocks per pixel – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | CPP_V | 00000010 | Sensor clocks per pixel, video mode |
| 8 - 15 | Reserved | 00000000 | Reserved |

## HBLANK_V – Horizontal Blanking Period, Video Mode

Address: 0x010a Block: 0x02 Offset: 0x0a

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HBLANK_V | | | | | | | |

**Horizontal blanking period – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | HBLANK_V | 00000000 | Horizontal blanking period, video mode. Units are Sensor clocks x 2. Legal values are 0 to 255. |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register writes to sensor register "HBLANK – Horizontal Blank" on page 292, when the module is in video mode.

## VBLANK_V – Vertical Blanking Period, Video Mode

Address: 0x010c Block: 0x02 Offset: 0x0c

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VBLANK_V | | | | | | | |

**Vertical blanking period – video mode**

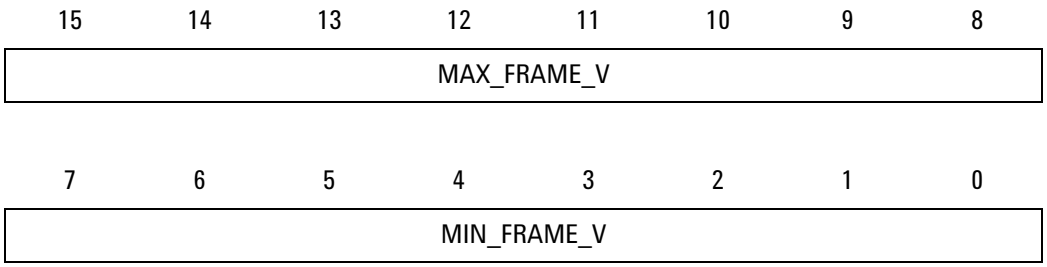
| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VBLANK_V | 00000000 | Vertical blanking period, video mode. Units are number of row processing periods. Legal values are 0 to 255. |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register writes to sensor register "VBLANK – Vertical Blank" on page 293, when the module is in video mode.

## MIN_MAX_F_V – Frame Convergence Rates, Video Mode

Address: 0x010e Block: 0x02 Offset: 0x0e

Access: Read / Write Reset value: 0x0000 Type: Two 8-bit integers




**Frame convergence rates – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | MIN_FRAME_V | 00000000 | Minimum number of frames before auto functions converge |
| 8 - 15 | MAX_FRAME_V | 00000000 | Maximum number of frames before auto functions converge |

**NOTE** This register is used for debugging. Setting the values to 0 disables the function. It is recommended these values NOT be changed.

## OUTPUT_CTRL_V – Output Control, Video Mode

Address: 0x0110 Block: 0x02 Offset: 0x10

Access: Read / Write Reset value: 0x9019 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| JUST_V | EOF_V | SOF_V | FR_OUT_V | ESYNC_V | 4BCCIR | HSMODE_V | FRVCLK_V |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PHYSNC_V | PVSYNC_V | PVCLK_V | 1 | OUTPUT_F_V | | | |

**Output control – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | OUTPUT_F_V | 1001 | Output format, video mode:<br>0000:RGB 888 (3 pixels in 3 bytes)<br>0001:RGB 666A tight pack (4 pixels in 9 bytes)<br>0010:RGB 666B loose pack (1 pixel in 3 bytes, left or right justified)<br>0011:RGB 565 (1 pixel in 2 bytes)<br>0100:RGB 444A tight pack (2 pixels in 3 bytes)<br>0101:RGB 444B loose pack (1 pixel in 2 bytes)<br>0110:RGB 444C sparse pack (1 pixel in 3 bytes)<br>0111:RGB 332 (1 pixel in 1 byte)<br>1000:YCbCr 4:2:2 A (Y1, Cb12, Y2, Cr12 order)<br>**1001:YCbCr 4:2:2 B (Cb12, Y1, Cr12, Y2 order)**<br>1010:YCbCr 4:2:2 C (Y1, Cr12, Y2, Cb12 order)<br>1011:YCbCr 4:2:2 D (Cr12, Y1, Cb12, Y2 order)<br>1100:4:4:4 YCbCr<br>1101:4:0:0 grayscale<br>1110:Raw – with AWB and BPA<br>1111:Raw – without AWB and BPA |
| 4 | Reserved | 1 | Reserved – note non-zero value |
| 5 | PVCLK_V | 0 | VCLK signal polarity:<br>**0: Data valid on rising edge**<br>1: Data valid on falling edge |
| 6 | PVSYNC_V | 0 | VSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |

**Output control – video mode (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 7 | PHSYNC_V | 0 | HSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |
| 8 | FRVCLK_V | 0 | Turn off free running VCLK:<br>**0: VCLK runs all the time the ipipe clock is running**<br>1: VCLK cycles once for each data byte output |
| 9 | HSMODE_V | 0 | HSYNC/VSYNC mode select:<br>**0: HSYNC high for entire line, VSYNC high for entire frame (HSYNC/VSYNC mode)**<br>1: HSYNC high for last pixel of a line, VSYNC high for the last pixel of a frame. EOL/EOF mode, HSYNC = EOL, VSYNC = EOF. (Not recommended for use with free running VCLK) |
| 10 | 4BCCIR | 0 | 4-bit CCIR mode:<br>**0: Disabled, 8-bit CCIR mode**<br>1: Enabled, 4-bit CCIR mode |
| 11 | ESYNC_V | 0 | CCIR embedded synchronization – use ONLY with output formats 8 - 11<br>MUST set R_Y_MAX_MIN, G_CB_MAX_MIN, B_CR_MAX_MIN:<br>**0: Disabled**<br>1: Enabled |
| 12 | FR_OUT_V | 1 | Frame output control:<br>0: Frame output controlled by CMD2[0]<br>**1: All frames output** |
| 13 | SOF_V | 0 | Use start of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 14 | EOF_V | 0 | Use end of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 15 | JUST_V | 1 | Output justification – if output pads with zeros:<br>0: Left justified<br>**1: Right justified** |

## PROC_CTRL_V – Processing Control, Video Mode

Address: 0x0112 Block: 0x02 Offset: 0x12

Access: Read / Write Reset value: 0x0280 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SSA | AVE | V_FLIP_V | H_FLIP_V | SHARP_V | | HALF_W_V | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTW_V | TMB_V | CBB_V | SB_V | DMB_V | | G1G2_V | BPA_V |

**Processing control – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | BPA_V | 0 | Bad pixel correction bypass:<br>**0: Disabled – BPA enabled**<br>1: Enabled – BPA disabled |
| 1 | G1G2_V | 0 | Green pixel filter disable:<br>**0: Filter on**<br>1: Filter off |
| 2 - 3 | DMB_V | 00 | Demosaic bypass:<br>**00: Use full demosaic**<br>01: Map pixel input to red; zero the green and blue outputs<br>10: Map pixel input to its correct color; zero other two colors<br>11: Superpixel demosaic |
| 4 | SB_V | 0 | Sizer bypass:<br>**0: Disabled – sizer enabled**<br>1: Enabled – sizer disabled |
| 5 | CBB_V | 0 | Color balance bypass:<br>**0: Use color balance matrix**<br>1: No color balance |
| 6 | TMB_V | 0 | Tonemap bypass:<br>**0: Use tonemap**<br>1: Tonemap disabled |
| 7 | TWW_V | 1 | Tonemap table weighting:<br>0: Linear spacing<br>**1: Bottom-weighted spacing** |

**Processing control – video mode (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 - 9 | HALF_W_V | 10 | Halftone output width:<br>00: 4-bit<br>01: 6-bit<br>**10: 8-bit – halftone bypassed**<br>11: Reserved |
| 10 - 11 | SHARP_V | 00 | Sharpening:<br>**00: No effect**<br>01: Low effect<br>10: Medium effect<br>11: High effect |
| 12 | H_FLIP_V | 0 | Horizontal flip (mirroring) – status only, see note below:<br>**0: Normal output**<br>1: Horizontally flipped output |
| 13 | V_FLIP_V | 0 | Vertical flip (mirroring) – status only, see note below:<br>**0: Normal output**<br>1: Vertically flipped output |
| 14 | AVE | 0 | Anti-vignetting enable<br>**0: Anti-vignetting disabled**<br>1: Anti-vignetting enabled |
| 15 | SSA | 0 | Sensor 2:1 subsampling active:<br>**0: Tells anti-vignetting that the sensor is not configured for subsampling**<br>1: Tells anti-vignetting that the sensor is configured for 2:1 subsampling |

**NOTE**

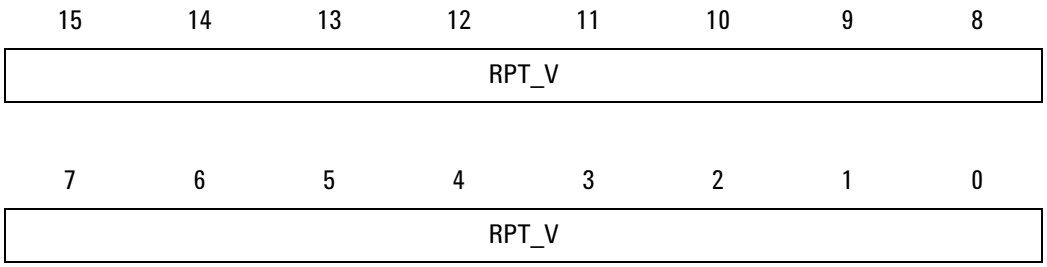
Horizontal and vertical mirroring for video mode should be done using the SIZE register, see page 145.

If the value of PROC_CTRL_V needs to be changed, read the value of the register first to determine the current state of H_FLIP_V and V_FLIP_V. These bit values MUST be written back exactly as read. If not, the color of the picture will be incorrect.

## RPT_V – Row Processing Time, Video Mode

Address: 0x0114 Block: 0x02 Offset: 0x14

Access: Read / Write Reset value: 0x0546 Type: 16-bit integer




**Row processing time – video mode**

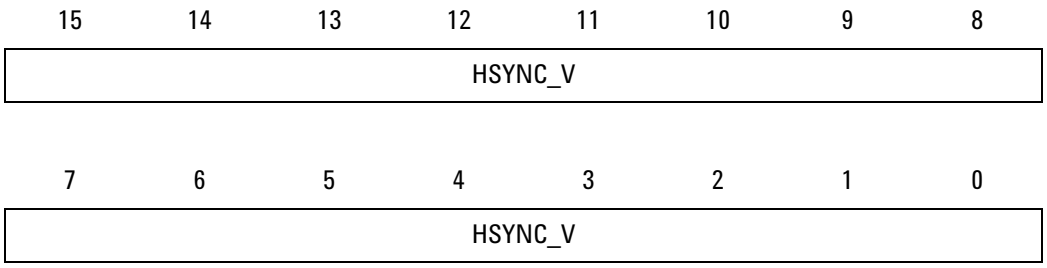
| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | RPT_V | 00000101<br>01000110 | Row processing time<br>Units are sensor clocks |

**NOTE** The value of this register is calculated by the camera controller, based on the current settings.

## HSYNC_PER_V – HSYNC Period, Video Mode

Address: 0x0116 Block: 0x02 Offset: 0x16

Access: Read / Write Reset value: 0x0a8b Type: 16-bit integer




**HSYNC period – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | HSYNC_V | 00001010<br>10001011 | Minimum number of HSYNC in image pipeline clocks for CCIR to process a line in video mode |

## CLK_DIV_V – Clock Divisors, Video Mode

Address: 0x0118 Block: 0x02 Offset: 0x18

Access: Read / Write Reset value: 0x0000 Type: Two 3-bit integers

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | IPIPE_CLK_V | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | SENS_CLK_V | | |

**Clock divisors – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | SENS_CLK_V | 000 | Sensor clock divisor – written to register SEN_CLK_DIV when in video mode |
| 3 - 7 | Reserved | 00000 | Reserved |
| 8 - 10 | IPIPE_CLK_V | 000 | Image pipeline clock divisor – written to register IP_CLK_DIV when in video mode |
| 11 - 15 | Reserved | 00000 | Reserved |

## PARALLEL_CTRL_V – Parallel Output Control, Video Mode

Address: 0x011a Block: 0x02 Offset: 0x1a

Access: Read / Write Reset value: 0x0003 Type: 8-bit integer, bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | DOLP_V | 0 | 0 | I_Y/R_V | I_Cb/G_V | I_Cr/B_V | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VCLK_DIV_V | | | | | | | |

**Parallel output control – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VCLK_DIV_V | 00000011 | VCLK divisor:<br>VCLK = (VCLK_DIV_V +1) image pipeline clocks per byte |
| 8 | Reserved | 0 | Reserved – resets to 0 |
| 9 | I_Cr/B_V | 0 | Invert MSB of Cr/B output:<br>**0: No**<br>1: Yes |
| 10 | I_Cb/G_V | 0 | Invert MSB of Cb/G output:<br>**0: No**<br>1: Yes |
| 11 | I_Y/R_V | 0 | Invert MSB of Y/R output:<br>**0: No**<br>1: Yes |
| 12 - 13 | Reserved | 00 | Reserved – resets to 0 |
| 14 | DOLP_V | 0 | Disable output line pacing:<br>**0: Output line pacing enabled**<br>1: Output line pacing disabled |
| 15 | Reserved | 00 | Reserved – resets to 0 |

## SEN_CTRL_V – Sensor Control, Video Mode

Address: 0x011c Block: 0x02 Offset: 0x1c

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | SUB_V |

**Sensor control – video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | SUB_V | 0 | Video subsampling:<br>0: Disabled<br>**1: Enabled** |
| 1 - 15 | Reserved | 00000000 0000000 | Reserved |

## RESERVED – Reserved

Address: 0x011e Block: 0x02 Offset: 0x1e

Access: Read / Write Reset value: 0x0000 Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to this register |

# Still Registers

## SZR_IN_WID_S – Sizer Input Width, Still Mode

Address: 0x0120 Block: 0x02 Offset: 0x20

Access: Read / Write Reset value: 0x0280 Type: 10-bit integer

Decimal: 640

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | I_WID_S | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| I_WID_S | | | | | | | |

**Sizer input width – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | I_WID_S | 10<br>10000000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE**

This register writes to image pipeline register "SZR_IN_W – Sizer Input Width" on page 352, when the module is in still mode.

## SZR_IN_HGT_S – Sizer Input Height, Still Mode

Address: 0x0122 Block: 0x02 Offset: 0x22

Access: Read / Write Reset value: 0x01e0 Type: 10-bit integer

Decimal: 480

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | I_HGT_S | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| I_HGT_S | | | | | | | |

**Sizer input height – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | I_HGT_S | 01 11100000 | Number of rows in one frame (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE**

This register writes to image pipeline register "SZR_IN_H – Sizer Input Height" on page 353, when the module is in still mode.

## SZR_OUT_WID_S – Sizer Output Width, Still Mode

Address: 0x0124 Block: 0x02 Offset: 0x24

Access: Read / Write Reset value: 0x0280 Type: 10-bit integer

Decimal: 640

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | O_WID_S | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| O_WID_S | | | | | | | |

**Sizer output width – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | O_WID_S | 10<br>10000000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE**

This register writes to image pipeline register "SZR_OUT_W – Sizer Output Width" on page 354, when the module is in still mode.

## SZR_OUT_HGT_S – Sizer Output Height, Still Mode

Address: 0x0126 Block: 0x02 Offset: 0x26

Access: Read / Write Reset value: 0x01e0 Type: 10-bit integer

Decimal: 480

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | O_HGT_S | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| O_HGT_S | | | | | | | |

**Sizer output height – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | O_HGT_S | 01 11100000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** This register writes to image pipeline register "SZR_OUT_H – Sizer Output Height" on page 355, when the module is in still mode.

## CPP_S – Clocks per Pixel, Still Mode

Address: 0x0128 Block: 0x02 Offset: 0x28

Access: Read / Write Reset value: 0x0002 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CPP_S | | | | | | | |

**Clocks per pixel – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | CPP_S | 00000010 | Sensor clocks per pixel, still mode |
| 8 - 15 | Reserved | 00000000 | Reserved |

## HBLANK_S – Horizontal Blanking Period, Still Mode

Address: 0x012a Block: 0x02 Offset: 0x2a

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HBLANK_S | | | | | | | |

**Horizontal blanking period – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | HBLANK_S | 00000000 | Horizontal blanking period, still mode. Units are sensor clocks x 2 and legal values are 0 to 255. |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register writes to sensor register "HBLANK – Horizontal Blank" on page 292, when the module is in still mode.

## VBLANK_S – Vertical Blanking Period, Still Mode

Address: 0x012c Block: 0x02 Offset: 0x2c

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VBLANK_S | | | | | | | |

**Vertical blanking period – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VBLANK_S | 00000000 | Vertical blanking period, still mode. Units are number of row processing periods and legal values are 0 to 255. |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register writes to sensor register "VBLANK – Vertical Blank" on page 293, when the module is in still mode.

## MIN_MAX_F_S – Frame Convergence Rates, Still Mode

Address: 0x012e Block: 0x02 Offset: 0x2e

Access: Read / Write Reset value: 0x0002 Type: Two 8-bit integers

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| MAX_FRAME_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MIN_FRAME_S | | | | | | | |

**Frame convergence rates – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | MIN_FRAME_S | 00000010 | Minimum number of frames before auto functions converge |
| 8 - 15 | MAX_FRAME_S | 00000000 | Maximum number of frames before auto functions converge |

**NOTE** This register is used for debugging. Setting the values to 0 disables the function. It is recommended that these values NOT be changed.

## OUTPUT_CTRL_S – Output Control, Still Mode

Address: 0x0130 Block: 0x02 Offset: 0x30

Access: Read / Write Reset value: 0x8019 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| JUST_S | EOF_S | SOF_S | FR_OUT_S | ESYNC_S | 4BCCIR | HSMODE_S | FRVCLK_S |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PHYSNC_S | PVSYNC_S | PVCLK_S | 1 | OUTPUT_F_S | | | |

**Output control – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | OUTPUT_F_S | 1001 | Output format:<br>0000:RGB 888 (3 pixels in 3 bytes)<br>0001:RGB 666A tight pack (4 pixels in 9 bytes)<br>0010:RGB 666B loose pack (1 pixel in 3 bytes, left or right justified)<br>0011:RGB 565 (1 pixel in 2 bytes)<br>0100:RGB 444A tight pack (2 pixels in 3 bytes)<br>0101:RGB 444B loose pack (1 pixel in 2 bytes)<br>0110:RGB 444C sparse pack (1 pixel in 3 bytes)<br>0111:RGB 332 (1 pixel in 1 byte)<br>1000:YCbCr 4:2:2 A (Y1, Cb12, Y2, Cr12 order)<br>**1001:YCbCr 4:2:2 B (Cb12, Y1, Cr12, Y2 order)**<br>1010:YCbCr 4:2:2 C (Y1, Cr12, Y2, Cb12 order)<br>1011:YCbCr 4:2:2 D (Cr12, Y1, Cb12, Y2 order)<br>1100:4:4:4 YCbCr<br>1101:4:0:0 grayscale<br>1110:Raw – with AWB and BPA<br>1111:Raw – without AWB and BPA |
| 4 | Reserved | 1 | Reserved – note non-zero value |
| 5 | PVCLK_S | 0 | VCLK signal polarity:<br>**0: Data valid on rising edge**<br>1: Data valid on falling edge |
| 6 | PVSYNC_S | 0 | VSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |

**Output control – still mode (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 7 | PHSYNC_S | 0 | HSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |
| 8 | FRVCLK_S | 0 | Turn off free running VCLK:<br>**0: VCLK runs all the time the ipipe clock is running**<br>1: VCLK cycles once for each data byte output |
| 9 | HSMODE_S | 0 | HSYNC/VSYNC mode select:<br>**0: HSYNC high for entire line, VSYNC high for entire frame (HSYNC/VSYNC mode)**<br>1: HSYNC high for the last pixel of a line, VSYNC high for the last pixel of a frame. EOL/EOF mode, HSYNC = EOL, VSYNC = EOF. (Not recommended for use with free running VCLK) |
| 10 | 4BCCIR | 0 | 4-bit CCIR mode:<br>**0: Disabled, 8-bit CCIR mode**<br>1: Enabled, 4-bit CCIR mode |
| 11 | ESYNC_S | 0 | CCIR embedded synchronization – use ONLY with output formats 8 - 11<br>MUST set R_Y_MAX_MIN, G_CB_MAX_MIN, B_CR_MAX_MIN:<br>**0: Disabled**<br>1: Enabled |
| 12 | FR_OUT_S | 0 | Frame output control:<br>**0: Frame output controlled by CMD2[0]**<br>1: All frames output |
| 13 | SOF_S | 0 | Use start of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 14 | EOF_S | 0 | Use end of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 15 | JUST_S | 1 | Output justification – if output pads with zeros:<br>0: Left justified<br>**1: Right justified** |

## PROC_CTRL_S – Processing Control, Still Mode

Address: 0x0132 Block: 0x02 Offset: 0x32

Access: Read / Write Reset value: 0x0280 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SSA | AVE | V_FLIP_S | H_FLIP_S | SHARP_S | | HALF_W_S | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTW_S | TMB_S | CBB_S | SB_S | DMB_S | | G1G2_S | BPA_S |

**Processing control – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | BPA_S | 0 | Bad pixel correction bypass:<br>**0: Disabled – BPA enabled**<br>1: Enabled – BPA disabled |
| 1 | G1G2_S | 0 | Green pixel filter disable:<br>**0: Filter on**<br>1: Filter off |
| 2 - 3 | DMB_S | 00 | Demosaic bypass:<br>**00: Use full demosaic**<br>01: Map pixel input to red; zero the green and blue outputs<br>10: Map pixel input to its correct color; zero other two colors<br>11: Superpixel demosaic |
| 4 | SB_S | 0 | Sizer bypass:<br>**0: Disabled – sizer enabled**<br>1: Enabled – sizer disabled |
| 5 | CBB_S | 0 | Color balance bypass:<br>**0: Use color balance matrix**<br>1: No color balance |
| 6 | TMB_S | 0 | Tonemap bypass:<br>**0: Use tonemap**<br>1: Tonemap disabled |
| 7 | TWW_S | 1 | Tonemap table weighting:<br>0: Linear spacing<br>**1: Bottom-weighted spacing** |

**Processing control – still mode (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 - 9 | HALF_W_S | 10 | Halftone output width:<br>00: 4-bit<br>01: 6-bit<br>**10: 8-bit – halftone bypassed**<br>11: Reserved |
| 10 - 11 | SHARP_S | 00 | Sharpening:<br>**00: No effect**<br>01: Low effect<br>10: Medium effect<br>11: High effect |
| 12 | H_FLIP_S | 0 | Horizontal flip (mirroring) – status only, see note below:<br>**0: Normal output**<br>1: Horizontally flipped output |
| 13 | V_FLIP_S | 0 | Vertical flip (mirroring) – status only, see note below:<br>**0: Normal output**<br>1: Vertically flipped output |
| 14 | AVE | 0 | Anti-vignetting enable<br>**0: Anti-vignetting disabled**<br>1: Anti-vignetting enabled |
| 15 | SSA | 0 | Sensor 2:1 subsampling active:<br>**0: Tells anti-vignetting that the sensor is not configured for subsampling**<br>1: Tells anti-vignetting that the sensor is configured for 2:1 subsampling |

**NOTE**

Horizontal and vertical mirroring for still mode should be done using the SIZE register, see page 145.

If the value of PROC_CTRL_S needs to be changed, first read the value of the register to determine the current state of the H_FLIP_S and V_FLIP_S registers. These bit values MUST be written back exactly as read. If not, the color of the picture will be incorrect.

## RPT_S – Row Processing Time, Still Mode

Address: 0x0134 Block: 0x02 Offset: 0x34

Access: Read / Write Reset value: 0x0546 Type: 16-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| RPT_S | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RPT_S | | | | | | | |

**Row processing time – still mode**

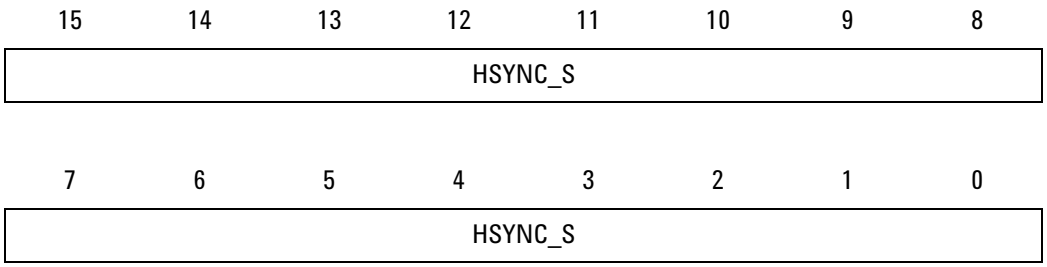
| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | RTP_S | 00000101<br>01000110 | Row processing time<br>Units are sensor clocks |

**NOTE** The value of this register is calculated by the camera controller, based on the current settings.

## HSYNC_PER_S – HSYNC Period, Still Mode

Address: 0x0136 Block: 0x02 Offset: 0x36

Access: Read / Write Reset value: 0x0545 Type: 16-bit integer




**HSYNC – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | HSYNC_S | 00000101<br>00100101 | Minimum number of HSYNC in image pipeline clocks for CCIR to process a line in still mode |

## CLK_DIV_S – Clock Divisors, Still Mode

Address: 0x0138 Block: 0x02 Offset: 0x38

Access: Read / Write Reset value: 0x0000 Type: Two 3-bit integers

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | IPIPE_CLK_S | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | SENS_CLK_S | | |

**Clock divisors – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | SENS_CLK_S | 000 | Sensor clock divisor – written to SEN_CLK_DIV when in still mode |
| 3 - 7 | Reserved | 00000 | Reserved |
| 8 - 10 | IPIPE_CLK_S | 000 | Image pipeline clock divisor – written to IP_CLK_DIV when in still mode |
| 11 - 15 | Reserved | 00000 | Reserved |

## PARALLEL_CTRL_S – Parallel Output Control, Still Mode

Address: 0x013a Block: 0x02 Offset: 0x3a

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer, bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | DOLP_S | 0 | 0 | I_Y/R_S | I_Cb/G_S | I_Cr/B_S | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VCLK_DIV_S | | | | | | | |

**Parallel output control – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VCLK_DIV_S | 00000000 | VCLK divisor:<br>VLCK = (VCLK_DIV_S +1) ipipe clocks per byte |
| 8 | Reserved | 0 | Reserved – resets to 0 |
| 9 | I_Cr/B_S | 0 | Invert MSB of Cr/B output:<br>**0: No**<br>1: Yes |
| 10 | I_Cb/G_S | 0 | Invert MSB of Cb/G output:<br>**0: No**<br>1: Yes |
| 11 | I_Y/R_S | 0 | Invert MSB of Y/R output:<br>**0: No**<br>1: Yes |
| 12 - 13 | Reserved | 00 | Reserved – resets to 0 |
| 14 | DOLP_S | 0 | Disable output line pacing:<br>**0: Output line pacing enabled**<br>1: Output line pacing disabled |
| 15 | Reserved | 0 | Reserved – resets to 0 |

## SEN_CTRL_S – Sensor Control, Still Mode

Address: 0x013c Block: 0x02 Offset: 0x1c

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | VF_S | HF_S | SUB_S |

**Sensor control – still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | SUB_S | 0 | Still subsampling:<br>0: Disabled<br>**1: Enabled** |
| 1 | HF_S | 0 | Still horizontal flip (mirroring):<br>0: Disabled<br>**1: Enabled** |
| 2 | VF_S | 0 | Still vertical flip (mirroring):<br>0: Disabled<br>**1: Enabled** |
| 3 - 15 | Reserved | 0000000000000 | Reserved |

**NOTE**

Horizontal and vertical flipping should be done using the SIZE register (see page 145). If flipping is done using these bits, the values will be overwritten the next time the CONFIG bit is set in the CONTROL register.

The HF_S and VF_S bits are effective immediately. The SUB_S bit does not take effect until the camera is stopped and then restarted. (SNAP bit in CONTROL)

## RESERVED – Reserved

Address: 0x013e Block: 0x02 Offset: 0x3e

Access: Read / Write Reset value: 0x0000 Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to this register |

# Auto Functions Control

## AF_CTRL1 – Auto Functions Control 1

Address: 0x0140 Block: 0x02 Offset: 0x40

Access: Read / Write Reset value: 0x0013 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | ABL | NACC | AFD | AWB | AE |

**Auto functions control 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | AE | 1 | Auto exposure:<br>0: Disabled<br>**1: Enabled** |
| 1 | AWB | 1 | Auto white balance:<br>0: Disabled<br>**1: Enabled** |
| 2 | AFD | 0 | Auto flicker detect:<br>**0: Disabled**<br>1: Enabled |
| 3 | NACC | 0 | Noise adaptive color correction:<br>**0: Disabled**<br>1: Enabled |
| 4 | ABL | 1 | Auto black level:<br>0: Disabled<br>**1: Enabled** |
| 5 - 15 | Reserved | 000 00000000 | Reserved |

## AF_CTRL2 – Auto Functions Control 2

Address: 0x0142 Block: 0x02 Offset: 0x42

Access: Read / Write Reset value: 0x0001 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | AFD_C_FR | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | RAF | ICONV | 0 | DOE_IFL | SR_D | AE50n60 | AEDOE |

**Auto functions control 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | AEDOE | 1 | Auto exposure deliberate overexposure:<br>0: Disabled<br>**1: Enabled** |
| 1 | AE50n60 | 0 | Auto exposure frequency:<br>**0: 50 Hz**<br>1: 60 Hz |
| 2 | SR_D | 0 | Sub row exposure disable:<br>**0: Use sub row exposure**<br>1: Sub row exposure disabled |
| 3 | DOE_IFL | 0 | Use DOE if flicker detected:<br>**0: Disabled**<br>1: Use DOE if flicker is detected |
| 4 | Reserved | 0 | Reserved |
| 5 | ICONV | 0 | Ignore auto functions convergence:<br>**0: Disabled**<br>1: Enabled |
| 6 | RAF | 0 | Re-initialize auto functions:<br>**0: Disabled**<br>1: Enabled |
| 7 - 10 | Reserved | 0000 | Reserved |
| 11 | AFD_C_FR | 0 | Change auto flicker detection frame rate:<br>**0: Disabled**<br>1: Enabled |
| 12 - 15 | Reserved | 0000 | Reserved |

## AF_STATUS – Auto Functions Status

Address: 0x0144 Block: 0x02 Offset: 0x44

Access: Read / Write Reset value: 0x0000 Type: Bit field

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | FS |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | AE_DOE | CC | DEF_F | DF | FD |

**Auto functions status**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | FD | 0 | Flicker detected:<br>**0: No**<br>1: Yes |
| 1 | DF | 0 | Detected flicker frequency:<br>**0: 50 Hz**<br>1: 60 Hz |
| 2 | DEF_F | 0 | Default flicker frequency:<br>**0: 50 Hz**<br>1: 60 Hz |
| 3 | CC | 0 | Camera auto functions converged:<br>**0: No**<br>1: Yes |
| 4 | AE_DOE | 0 | Auto exposure deliberate overexposure:<br>**0: Not active**<br>1: Active |
| 5 - 7 | Reserved | 000 | Reserved |
| 8 | FS | 0 | Flicker sticky bit – flicker detected since last reset:<br>**0: No**<br>1: Yes |
| 9 - 15 | Reserved | 0000000 | Reserved |

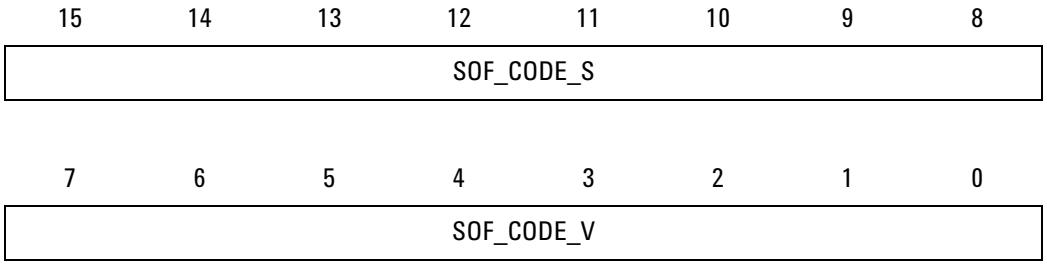
**NOTE** The value will change after the first frame. Usually the CC bit will be set.

# Start of Frame and End of Frame Codes

## SOF_CODES – Start of Frame Codes

Address: 0x0146 Block: 0x02 Offset: 0x46

Access: Read / Write Reset value: 0xfeff Type: Two 8-bit integers




**Start of frame codes**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SOF_CODE_V | 11111111 | Start of frame code, video mode |
| 8 - 15 | SOF_CODE_S | 11111110 | Start of frame code, still mode |

## EOF_CODES – End of Frame Codes

Address: 0x0148 Block: 0x02 Offset: 0x48

Access: Read / Write Reset value: 0x0100 Type: Two 8-bit integers

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| BAD_EOF | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GOOD_EOF | | | | | | | |

**End of frame codes**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | GOOD_EOF | 00000000 | Good end of frame code |
| 8 - 15 | BAD_EOF | 00000001 | Bad end of frame code |

# Auto Black Level

## ABL_TARGET – Auto Black Level Target

Address: 0x014a Block: 0x02 Offset: 0x4a

Access: Read / Write Reset value: 0x0005 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ABL_TGT | | | | | | | |

**Auto black level target**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | ABL_TGT | 00000101 | Auto black level target – units are 0.1% |
| 8 - 15 | Reserved | 00000000 | Reserved |

## ABL_MAX_BIN – Auto Black Level Maximum Bin Number

Address: 0x014c Block: 0x02 Offset: 0x4c

Access: Read / Write Reset value: 0x0003 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ABL_M_BIN | | | | | | | |

**Auto black level maximum bin number**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | ABL_M_BIN | 00000011 | Auto black level target maximum bin number |
| 8 - 15 | Reserved | 00000000 | Reserved |

## ABL_MAX_BLACK – Auto Black Level Maximum Black

Address: 0x014e Block: 0x02 Offset: 0x4e

Access: Read / Write Reset value: 0x0010 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ABL_M_BLACK | | | | | | | |

**Auto black level maximum black**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | ABL_M_BLACK | 00000011 | Auto black level target maximum black |
| 8 - 15 | Reserved | 00000000 | Reserved |

# Auto Exposure

## AE_GAIN_MIN – Auto Exposure Gain Minimum

Address: 0x0150 Block: 0x02 Offset: 0x50

Access: Read / Write Reset value: 0x01c0 Type: Real (8.8 format)

Decimal: 1.75

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_MIN_G | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_MIN_G | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure gain absolute minimum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Absolute minimum gain; specifies the minimum gain level used by the auto exposure function |
| 0 - 7 | AE_MIN_G | 11000000 | Auto exposure absolute minimum gain, fractional portion, divide by 256 |
| 8 - 15 | | 00000001 | Auto exposure absolute minimum gain, integer portion |

## AE_GAIN_MIN_P – Auto Exposure Gain Minimum Preferred

Address: 0x0152 Block: 0x02 Offset: 0x52

Access: Read / Write Reset value: 0x0200 Type: Real (8.8 format)

Decimal: 2.0

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AEP_MIN_G | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AEP_MIN_G | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure gain preferred minimum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Preferred minimum gain. Specifies a preferred minimum gain level for the auto exposure function. To avoid lowering the gain below the preferred minimum level, the AE function will first decrease the exposure time until the minimum exposure time is reached. |
| 0 - 7 | AEP_MIN_G | 00000000 | Auto exposure preferred minimum gain, fractional portion, divide by 256 |
| 8 - 15 | | 00000010 | Auto exposure preferred minimum gain, integer portion |

## AE_GAIN_MAX – Auto Exposure Gain Maximum

Address: 0x0154 Block: 0x02 Offset: 0x54

Access: Read / Write Reset value: 0x0500 Type: Real (8.8 format)

Decimal: 5.0

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_MAX_G | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_MAX_G | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure gain maximum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Absolute maximum gain; specifies the maximum gain level used by the auto exposure function |
| 0 - 7 | AE_MAX_G | 00000000 | Auto exposure maximum gain, fractional portion, divide by 256 |
| 8 - 15 | | 00000101 | Auto exposure maximum gain, integer portion |

## AE_GAIN_DFLT – Auto Exposure Gain Default

Address: 0x0156 Block: 0x02 Offset: 0x56

Access: Read / Write Reset value: 0x0200 Type: Real (8.8 format)

Decimal: 2.0

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_DFLT_G | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_DFLT_G | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure gain default**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Default gain; specifies the gain first used by the auto exposure function |
| 0 - 7 | AE_DFLT_G | 00000000 | Auto exposure gain default, fractional portion, divide by 256 |
| 8 - 15 | | 00000010 | Auto exposure gain default, integer portion |

## AE_ETIME_MIN – Auto Exposure Time Minimum

Address: 0x0158 Block: 0x02 Offset: 0x58

Access: Read / Write Reset value: 0x0005 Type: 16-bit integer

Decimal: 50 µS

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_MIN | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_MIN | | | | | | | |

**Auto exposure time minimum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | AE_ETIME_MIN | 00000000 00000101 | Minimum exposure time; units are 10 µS |

## AE_ETIME_MAX – Auto Exposure Time Maximum

Address: 0x015a Block: 0x02 Offset: 0x5a

Access: Read / Write Reset value: 0x4e20 Type: 16-bit integer

Decimal: 200.00 mS

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_MAX | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_MAX | | | | | | | |

**Auto exposure time maximum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | AE_ETIME_MAX | 01001110 00100000 | Maximum exposure time; units are 10 µS |

**NOTE** The default value for this register will result in a frame rate of 5 frames/second when the camera is under very low light conditions. If a 15 frames/second minimum frame rate is needed, set the AE_ETIME_MAX bit to 0x1a0a.

## AE_ETIME_DFLT – Auto Exposure Time Default

Address: 0x015c Block: 0x02 Offset: 0x5c

Access: Read / Write Reset value: 0x03e8 Type: 16-bit integer

Decimal: 10.0 mS

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_DFLT | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_ETIME_DFLT | | | | | | | |

**Auto exposure time default**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | AE_ETIME_DFLT | 00000011 11101000 | Default exposure time; units are 10 µS |

## AE_TARGET – Auto Exposure Target

Address: 0x015e Block: 0x02 Offset: 0x5e

Access: Read / Write Reset value: 0x0040 Type: 8-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_TARGET | | | | | | | |

**Auto exposure target**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | AE_TARGET | 000100 | Auto exposure target; specifies the target image illuminance (average pixel intensity) in a range of 0 – 255 |
| 8 - 15 | Reserved | 000 | Reserved |

## AE_TOL_ACQ – Auto Exposure Tolerance Acquire

Address: 0x0160 Block: 0x02 Offset: 0x60

Access: Read / Write Reset value: 0x0118 Type: Real (8.8 format)

Decimal: 1.09375

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_TOL_ACQ | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_TOL_ACQ | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure tolerance acquire**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Auto exposure tolerance acquire – defines how close the auto exposure must be to the optimum value to be considered converged |
| 0 - 7 | AE_TOL_ACQ | 00011000 | Auto exposure tolerance acquire, fractional portion, divide by 256 |
| 8 - 15 | | 00000001 | Auto exposure tolerance acquire, integer portion |

**NOTE** Minimum value is 1.0 (0x0100).

## AE_TOL_MON – Auto Exposure Tolerance Monitor

Address: 0x0162 Block: 0x02 Offset: 0x62

Access: Read / Write Reset value: 0x0118 Type: Real (8.8 format)

Decimal: 1.09375

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_TOL_MON | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_TOL_MON | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure tolerance monitor**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Auto exposure tolerance monitor – defines how far the optimum exposure is from the current exposure before the auto exposure algorithm changes the exposure |
| 0 - 7 | AE_TOL_MON | 00011000 | Auto exposure tolerance monitor, fractional portion, divide by 256 |
| 8 - 15 | | 00000001 | Auto exposure tolerance monitor, integer portion |

**NOTE** Minimum value is 1.0 (0x0100).

## AE_MARGIN – Auto Exposure Margin

Address: 0x0164 Block: 0x02 Offset: 0x64

Access: Read / Write Reset value: 0x0120 Type: 16-bit real number (8.8 format)

Decimal: 1.125

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_MARGIN | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_MARGIN | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure margin**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Auto exposure margin – hysteresis for changing exposure times |
| 0 - 7 | AE_MARGIN | 00100000 | Auto exposure margin; fractional portion, divide by 256 |
| 8 - 15 | | 00000001 | Auto exposure margin, integer portion |

**NOTE** The minimum value is 1.0 (0x0100).

## AE_DOE_FACTOR – Auto Exposure Deliberate Overexposure Factor

Address: 0x0166 Block: 0x02 Offset: 0x66

Access: Read / Write Reset value: 0x014e Type: 16-bit real number (8.8 format)

Decimal: 1.5

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_DOE_FACTOR | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_DOE_FACTOR | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure DOE factor**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Deliberate overexposure factor is how overexposed the image can be before the auto exposure algorithm abandons flicker free exposure times |
| 0 - 7 | AE_DOE_FACTOR | 01001110 | Auto exposure DOE factor, fractional portion |
| 8 - 15 | | 00000001 | Auto exposure DOE factor, integer portion |

**NOTE** The minimum value is 1.0 (0x0100).

## AE_DOE_MARGIN – Auto Exposure Deliberate Overexposure Margin

Address: 0x0168 Block: 0x02 Offset: 0x68

Access: Read / Write Reset value: 0x0140 Type: 16-bit real number (8.8 format)

Decimal: 1.25

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AE_DOE_MARGIN | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AE_DOE_MARGIN | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto exposure DOE margin**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Deliberate overexposure margin – once DOE is enabled, margin of the hysteresis |
| 0 - 7 | AE_DOE_MARGIN | 01000000 | Auto exposure DOE margin, fractional portion |
| 8 - 15 | | 00000001 | Auto exposure DOE margin, integer portion |

**NOTE** The minimum value is 1.0 (0x0100).

## RESERVED – Reserved

Address: 0x016a 0x016c 0x016e  Block: 0x02  Offset: 0x6a 0x6c 0x6e

Access: Read / Write  Reset value: 0x0000  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to these registers |

# Auto White Balance

## AWB_RED_MIN – AWB Minimum Red/Green Ratio

Address: 0x0170 Block: 0x02 Offset: 0x70

Access: Read / Write Reset value: 0x00c0 Type: Real (3.8 format)

Decimal: 0.75

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | RED_MIN | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RED_MIN | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB minimum red/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | RED_MIN | 11000000 | Fractional part of the red/green ratio. Data in bits 0 - 7 are divided by 56, resulting in the decimal part of the gain. |
| 8 - 10 | | 000 | Integer part of the red/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_RED_MAX – AWB Maximum Red/Green Ratio

Address: 0x0172 Block: 0x02 Offset: 0x72

Access: Read / Write Reset value: 0x01a6 Type: Real (3.8 format)

Decimal: 1.6484375

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | RED_MAX | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RED_MAX | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB maximum red/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | RED_MAX | 10100110 | Fractional part of the red/green ratio. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the gain. |
| 8 - 10 | | 001 | Integer part of the red/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_RED_DFLT – AWB Default Red/Green Ratio

Address: 0x0174 Block: 0x02 Offset: 0x74

Access: Read / Write Reset value: 0x0134 Type: Real (3.7 format)

Decimal: 1.203125

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | RED_DFLT | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RED_DFLT | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB default red/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | RED_DFLT | 00110100 | Fractional part of the red/green ratio. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the gain. |
| 8 - 10 | | 001 | Integer part of the red/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_BLUE_MIN – AWB Minimum Blue/Green Ratio

Address: 0x0176 Block: 0x02 Offset: 0x76

Access: Read / Write Reset value: 0x00c0 Type: Real (3.8 format)

Decimal: 0.75

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | BLUE_MIN | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BLUE_MIN | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB minimum blue/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | BLUE_MIN | 11000000 | Fractional part of the blue/green ratio. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the gain. |
| 8 - 10 | | 000 | Integer part of the blue/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_BLUE_MAX – AWB Maximum Blue/Green Ratio

Address: 0x0178 Block: 0x02 Offset: 0x78

Access: Read / Write Reset value: 0x02a4 Type: Real (3.8 format)

Decimal: 2.640625

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | BLUE_MAX | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BLUE_MAX | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB maximum blue/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | BLUE_MAX | 10100100 | Fractional part of the blue/green ratio. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the gain. |
| 8 - 10 | | 010 | Integer part of the blue/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_BLUE_DFLT – AWB Default Blue/Green Ratio

Address: 0x017a Block: 0x02 Offset: 0x7a

Access: Read / Write Reset value: 0x01e4 Type: Real (3.8 format)

Decimal: 1.890625

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | BLUE_DFLT | | |
| Reserved | | | | | Integer part of number | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BLUE_DFLT | | | | | | | |
| Fractional part of number | | | | | | | |

**AWB default blue/green ratio**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | BLUE_DFLT | 11100100 | Fractional part of the blue/green ratio. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the gain. |
| 8 - 10 | | 001 | Integer part of the blue/green ratio |
| 11 - 15 | Reserved | 00000 | Reserved |

## AWB_TOL_ACQ – Auto White Balance Tolerance Acquire

Address: 0x017c Block: 0x02 Offset: 0x7c

Access: Read / Write Reset value: 0x0110 Type: Real (8.8 format)

Decimal: 1.0625

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AWB_TOL_ACQ | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AWB_TOL_ACQ | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto white balance tolerance acquire**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Auto white balance tolerance acquire – defines how close the current white balance is to the optimum white balance before the auto white balance algorithm is considered converged |
| 0 - 7 | AWB_TOL_ACQ | 00010000 | Fractional part of the AWB tolerance acquire. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the tolerance. |
| 8 - 15 | | 00000001 | Integer part of the AWB tolerance acquire. |

## AWB_TOL_MON – Auto White Balance Tolerance Monitor

Address: 0x017e Block: 0x02 Offset: 0x7e

Access: Read / Write Reset value: 0x0120 Type: Real (8.8 format)

Decimal: 1.125

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| AWB_TOL_MON | | | | | | | |
| Integer part of number | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AWB_TOL_MON | | | | | | | |
| Fractional part of number | | | | | | | |

**Auto white balance tolerance monitor**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Auto white balance tolerance monitor – defines how far the optimum white balance is from the current white balance before the auto white balance algorithm changes the white balance. |
| 0 - 7 | AWB_TOL_MON | 00100000 | Fractional part of the AWB tolerance monitor. Data in bits 0 - 7 are divided by 256, resulting in the decimal part of the tolerance. |
| 8 - 15 | | 00000001 | Integer part of the AWB tolerance monitor |

# Current Firmware Revision

## FIRMWARE_REV – Current Firmware Revision

Address: 0x0180 Block: 0x03 Offset: 0x00

Access: Read / Write Reset value: 0x0152 Type: BCD number

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| FW_REV | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FW_REV | | | | | | | |

**Firmware revision**

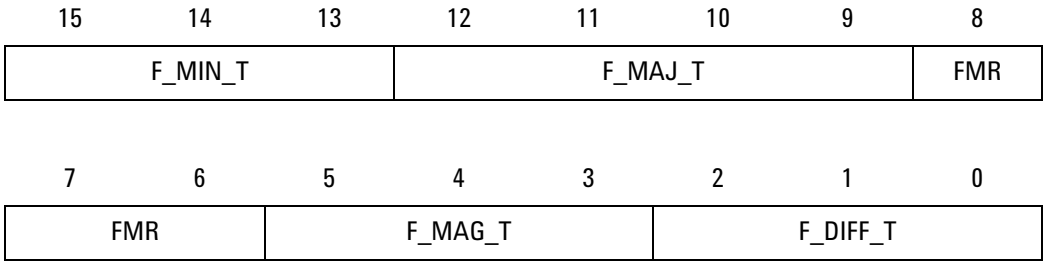
| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | FW_REV | 00000001<br>01010010 | Current firmware revision |

# Flicker Configuration

## FLICK_CFG_1 – Flicker Configuration 1

Address: 0x0182 Block: 0x03 Offset: 0x02

Access: Read / Write Reset value: 0x2aeb Type: Four 3-bit numbers, one 4-bit number




**Flicker configuration 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | F_DIFF_T | 011 | Flicker difference threshold:<br>000 = 2<br>001 = 4<br>010 = 7<br>**011 = 11**<br>100 = 16<br>101 = 24<br>110 = 40<br>111 = 255 (off) |
| 3 - 5 | F_MAG_T | 101 | Flicker magnitude threshold:<br>000 = 0.0 (off)<br>001 = 0.05 Cpp<br>010 = 0.1 Cpp<br>011 = 0.2 Cpp<br>100 = 0.5 Cpp<br>**101 = 1.0 Cpp**<br>110 = 2.0 Cpp<br>111 = 5.0 Cpp |
| 6 - 8 | FMR | 011 | Flicker magnitude ratio:<br>000 = 1.0<br>001 = 1.25<br>010 = 1.6<br>**011 = 2.0**<br>100 = 2.5<br>101 = 3.3<br>110 = 5.0<br>111 = 8.0 |
| 9 - 12 | F_MAJ_T | 0101 | Flicker majority threshold; the majority frequency must be detected this many times in the last 16 attempts before it is selected |
| 13 - 15 | F_MIN_T | 001 | Flicker minority threshold; If the minority frequency is detected at least this many times in the last 16 attempts AFD will decide there is no flicker |

## FLICK_CFG_2 – Flicker Configuration 2

Address: 0x0184 Block: 0x03 Offset: 0x04

Access: Read / Write Reset value: 0x0005 Type: 4-bit integer

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | INT | | | |

**Flicker configuration 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | INT | 0101 | Number of iterations that are used by the AFD to determine flicker |
| 4 - 15 | Reserved | 0000000000000000 | Reserved |

## RESERVED – Reserved

Address: 0x0186 0x0188 Block: 0x03 Offset: 0x06 0x08

Access: Read / Write Reset value: 0x0000 Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to these registers |

## MAX_SCLK – Maximum Sensor Clock

Address: 0x018a Block: 0x03 Offset: 0x0a

Access: Read / Write Reset value: 0x1964 Type: –

Decimal 6500 KHz

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| MAX_SCLK | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MAX_SCLK | | | | | | | |

**Maximum sensor clock**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | MAX_SCLK | 00011001 01100100 | Maximum sensor clock frequency. Units are in KHz; Default is 6.5 MHz |

## RESERVED – Reserved

Address: 0x018c 0x018e  Block: 0x03  Offset: 0x0c 0x0e

Access: Read / Write  Reset value: 0x0000  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | 00000000 00000000 | Reserved – DO NOT write any data to these registers |

# Color Space Conversion

## Color Space Conversion Coefficients, Video

Address: See table Block: 0x03 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit signed 2's complement (2.7 format)

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_xx_V |
| Reserved | | | | | | | Integer |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_xx_V | | | | | | | |
| Integer | Fractional part of number | | | | | | |

**Color space conversion coefficients, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | CSC_xx_V | See table | Fractional part of the coefficient; data in bits 0 - 6 are divided by 128, resulting in the decimal fraction of the coefficient |
| 7 - 8 | | See table | Signed 2's complement integer part of the coefficient |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion coefficient registers**

| | | | | Reset Value | | Binary | |
|---|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Hex | Decimal | Integer | Fractional |
| CSC_00_V | 0x0190 | 0x03 | 0x10 | 0x0026 | 0.296875 | 00 | 0100110 |
| CSC_01_V | 0x0192 | 0x03 | 0x12 | 0x004b | 0.5859375 | 00 | 1001011 |
| CSC_02_V | 0x0194 | 0x03 | 0x14 | 0x000f | 0.1171875 | 00 | 0001111 |
| CSC_10_V | 0x0196 | 0x03 | 0x16 | 0x01ed | – 0.1584375 | 11 | 1101101 |
| CSC_11_V | 0x0198 | 0x03 | 0x18 | 0x01db | – 0.2890625 | 11 | 1011011 |
| CSC_12_V | 0x019a | 0x03 | 0x1a | 0x0038 | 0.4375 | 00 | 0111000 |
| CSC_20_V | 0x019c | 0x03 | 0x1c | 0x004f | 0.6171875 | 00 | 1001111 |
| CSC_21_V | 0x019e | 0x03 | 0x1e | 0x01be | – 0.515625 | 11 | 0111110 |
| CSC_22_V | 0x01a0 | 0x03 | 0x20 | 0x01f3 | – 0.1015625 | 11 | 1110011 |

## Color Space Conversion Offsets, Video

Address: See table Block: 0x03 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit 2's complement number

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_OSx_V |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_OSx_V | | | | | | | |

**Color space conversion offsets, video mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | CSC_OSx_V | See table | Integer offset value |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion offset registers**

| | | | | Reset Value | | |
|---|---|---|---|---|---|---|
| **Register name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** | **Binary** |
| CSC_OS0_V | 0x01a2 | 0x03 | 0x22 | 0x0000 | 0 | 000000000 |
| CSC_OS1_V | 0x01a4 | 0x03 | 0x24 | 0x0080 | 128 | 010000000 |
| CSC_OS2_V | 0x01a6 | 0x03 | 0x26 | 0x0080 | 128 | 010000000 |

## Color Space Conversion Coefficients, Still

Address: See table Block: 0x03 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit signed 2's complement (2.7 format)

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_xx_S |
| Reserved | | | | | | | Integer |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_xx_S | | | | | | | |
| Integer | Fractional part of number | | | | | | |

**Color space conversion coefficients, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | CSC_xx_S | See table | Fractional part of the coefficient; data in bits 0 - 6 are divided by 128, resulting in the decimal fraction of the coefficient |
| 7 - 8 | | See table | Signed 2's complement integer part of the coefficient |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion coefficient registers**

| | | | | Reset Value | | Binary | |
|---|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Hex | Decimal | Integer | Fractional |
| CSC_00_S | 0x01a8 | 0x03 | 0x28 | 0x0026 | 0.296875 | 00 | 0100110 |
| CSC_01_S | 0x01aa | 0x03 | 0x2a | 0x004b | 0.5859375 | 00 | 1001011 |
| CSC_02_S | 0x01ac | 0x03 | 0x2c | 0x000f | 0.1171875 | 00 | 0001111 |
| CSC_10_S | 0x01ae | 0x03 | 0x2e | 0x01ed | – 0.1584375 | 11 | 1101101 |
| CSC_11_S | 0x01b0 | 0x03 | 0x30 | 0x01db | – 0.2890625 | 11 | 1011011 |
| CSC_12_S | 0x01b2 | 0x03 | 0x32 | 0x0038 | 0.4375 | 00 | 0111000 |
| CSC_20_S | 0x01b4 | 0x03 | 0x34 | 0x004f | 0.6171875 | 00 | 1001111 |
| CSC_21_S | 0x01b6 | 0x03 | 0x36 | 0x01be | – 0.515625 | 11 | 0111110 |
| CSC_22_S | 0x01b8 | 0x03 | 0x38 | 0x01f3 | – 0.1015625 | 11 | 1110011 |

## Color Space Conversion Offsets, Still

Address: See table Block: 0x03 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit 2's complement number

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_OSx_S |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_OSx_S | | | | | | | |

**Color space conversion offsets, still mode**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | CSC_OSx_S | See table | Integer offset value |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion offset registers**

| | | | | Reset Value | | |
|---|---|---|---|---|---|---|
| **Name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** | **Binary** |
| CSC_OS0_S | 0x01ba | 0x03 | 0x3a | 0x0000 | 0 | 000000000 |
| CSC_OS1_S | 0x01bc | 0x03 | 0x3c | 0x0080 | 128 | 010000000 |
| CSC_OS2_S | 0x01be | 0x03 | 0x3e | 0x0080 | 128 | 010000000 |

# Gamma (Tonemap)

The following coefficients, for video and still modes, are written out to the tonemap RAM locations. Access is Read/Write.

## Tonemap Coefficients

Address: See table Block: 0x03 Offset: See table

Access: Read / Write Reset value: See table Type: 10-bit integer, last TM entry is 11 bits

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | TM_COEF_xx_x | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TM_COEF_xx_x | | | | | | | |

**Tonemap coefficients**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | TM_COEF_xx_x | See table | Tonemap coefficient value |
| 10 - 15 | Reserved | 000000 | Reserved |

**Tonemap coefficients, video mode**

| | | | | Reset Value | |
|---|---|---|---|---|---|
| **Name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** |
| TM_COEF_00_V | 0x01c0 | 0x03 | 0x40 | 0x0000 | 0 |
| TM_COEF_01_V | 0x01c2 | 0x03 | 0x42 | 0x0017 | 23 |
| TM_COEF_02_V | 0x01c4 | 0x03 | 0x44 | 0x0032 | 50 |
| TM_COEF_03_V | 0x01c6 | 0x03 | 0x46 | 0x0046 | 70 |
| TM_COEF_04_V | 0x01c8 | 0x03 | 0x48 | 0x0056 | 86 |
| TM_COEF_05_V | 0x01ca | 0x03 | 0x4a | 0x0064 | 100 |
| TM_COEF_06_V | 0x01cc | 0x03 | 0x4c | 0x0071 | 113 |
| TM_COEF_07_V | 0x01ce | 0x03 | 0x4e | 0x007c | 124 |
| TM_COEF_08_V | 0x01d0 | 0x03 | 0x50 | 0x0086 | 134 |
| TM_COEF_09_V | 0x01d2 | 0x03 | 0x52 | 0x0099 | 153 |
| TM_COEF_10_V | 0x01d4 | 0x03 | 0x54 | 0x00a9 | 169 |
| TM_COEF_11_V | 0x01d6 | 0x03 | 0x56 | 0x00b8 | 184 |
| TM_COEF_12_V | 0x01d8 | 0x03 | 0x58 | 0x00c6 | 198 |
| TM_COEF_13_V | 0x01da | 0x03 | 0x5a | 0x00df | 223 |
| TM_COEF_14_V | 0x01dc | 0x03 | 0x5c | 0x00f5 | 245 |
| TM_COEF_15_V | 0x01de | 0x03 | 0x5e | 0x0109 | 265 |
| TM_COEF_16_V | 0x01e0 | 0x03 | 0x60 | 0x011b | 283 |
| TM_COEF_17_V | 0x01e2 | 0x03 | 0x62 | 0x013d | 317 |
| TM_COEF_18_V | 0x01e4 | 0x03 | 0x64 | 0x015a | 346 |
| TM_COEF_19_V | 0x01e6 | 0x03 | 0x66 | 0x0175 | 373 |
| TM_COEF_20_V | 0x01e8 | 0x03 | 0x68 | 0x018d | 397 |
| TM_COEF_21_V | 0x01ea | 0x03 | 0x6a | 0x01ba | 442 |
| TM_COEF_22_V | 0x01ec | 0x03 | 0x6c | 0x01e1 | 481 |
| TM_COEF_23_V | 0x01ee | 0x03 | 0x6e | 0x0205 | 517 |
| TM_COEF_24_V | 0x01f0 | 0x03 | 0x70 | 0x0225 | 549 |
| TM_COEF_25_V | 0x01f2 | 0x03 | 0x72 | 0x0261 | 609 |
| TM_COEF_26_V | 0x01f4 | 0x03 | 0x74 | 0x0295 | 661 |
| TM_COEF_27_V | 0x01f6 | 0x03 | 0x76 | 0x02c5 | 709 |
| TM_COEF_28_V | 0x01f8 | 0x03 | 0x78 | 0x02f1 | 753 |
| TM_COEF_29_V | 0x01fa | 0x03 | 0x7a | 0x033f | 831 |
| TM_COEF_30_V | 0x01fc | 0x03 | 0x7c | 0x0385 | 901 |
| TM_COEF_31_V | 0x01fe | 0x03 | 0x7e | 0x03c5 | 965 |
| TM_COEF_32_V | 0x0200 | 0x04 | 0x00 | 0x0400 | 1024 |

**Tonemap coefficients, still mode**

| | | | | Reset Value | |
|---|---|---|---|---|---|
| **Name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** |
| TM_COEF_00_S | 0x0202 | 0x04 | 0x02 | 0x0000 | 0 |
| TM_COEF_01_S | 0x0204 | 0x04 | 0x04 | 0x0017 | 23 |
| TM_COEF_02_S | 0x0206 | 0x04 | 0x06 | 0x0032 | 50 |
| TM_COEF_03_S | 0x0208 | 0x04 | 0x08 | 0x0046 | 70 |
| TM_COEF_04_S | 0x020a | 0x04 | 0x0a | 0x0056 | 86 |
| TM_COEF_05_S | 0x020c | 0x04 | 0x0c | 0x0064 | 100 |
| TM_COEF_06_S | 0x020e | 0x04 | 0x0e | 0x0071 | 113 |
| TM_COEF_07_S | 0x0210 | 0x04 | 0x10 | 0x007c | 124 |
| TM_COEF_08_S | 0x0212 | 0x04 | 0x12 | 0x0086 | 134 |
| TM_COEF_09_S | 0x0214 | 0x04 | 0x14 | 0x0099 | 153 |
| TM_COEF_10_S | 0x0216 | 0x04 | 0x16 | 0x00a9 | 169 |
| TM_COEF_11_S | 0x0218 | 0x04 | 0x18 | 0x00b8 | 184 |
| TM_COEF_12_S | 0x021a | 0x04 | 0x1a | 0x00c6 | 198 |
| TM_COEF_13_S | 0x021c | 0x04 | 0x1c | 0x00df | 223 |
| TM_COEF_14_S | 0x021e | 0x04 | 0x1e | 0x00f5 | 245 |
| TM_COEF_15_S | 0x0220 | 0x04 | 0x20 | 0x0109 | 265 |
| TM_COEF_16_S | 0x0222 | 0x04 | 0x22 | 0x011b | 283 |
| TM_COEF_17_S | 0x0224 | 0x04 | 0x24 | 0x013d | 317 |
| TM_COEF_18_S | 0x0226 | 0x04 | 0x26 | 0x015a | 346 |
| TM_COEF_19_S | 0x0228 | 0x04 | 0x28 | 0x0175 | 373 |
| TM_COEF_20_S | 0x022a | 0x04 | 0x2a | 0x018d | 397 |
| TM_COEF_21_S | 0x022c | 0x04 | 0x2c | 0x01ba | 442 |
| TM_COEF_22_S | 0x022e | 0x04 | 0x2e | 0x01e1 | 481 |
| TM_COEF_23_S | 0x0230 | 0x04 | 0x30 | 0x0205 | 517 |
| TM_COEF_24_S | 0x0232 | 0x04 | 0x32 | 0x0225 | 549 |
| TM_COEF_25_S | 0x0234 | 0x04 | 0x34 | 0x0261 | 609 |
| TM_COEF_26_S | 0x0236 | 0x04 | 0x36 | 0x0295 | 661 |
| TM_COEF_27_S | 0x0238 | 0x04 | 0x38 | 0x02c5 | 709 |
| TM_COEF_28_S | 0x023a | 0x04 | 0x3a | 0x02f1 | 753 |
| TM_COEF_29_S | 0x023c | 0x04 | 0x3c | 0x033f | 831 |
| TM_COEF_30_S | 0x023e | 0x04 | 0x3e | 0x0385 | 901 |
| TM_COEF_31_S | 0x0240 | 0x04 | 0x40 | 0x03c5 | 965 |
| TM_COEF_32_S | 0x0242 | 0x04 | 0x42 | 0x0400 | 1024 |

## RESERVED – Reserved

Address: 0x0244 ⋮ 0x024e  Block: 0x04  Offset: 0x44 ⋮ 0x0e

Access: Read / Write  Reset value: –  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | – | Reserved – DO NOT write any data to these registers |

# Noise Adaptive Color Correction

Address: See table Block: 0x04 Offset: See table

Access: Read / Write Reset value: See table Type: 16-bit integer

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| NACC_xxx_x | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| NACC_xxx_x | | | | | | | |

**Noise adaptive color correction**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | NACC_xxx_x | See table | NACC coefficients |

## Color Correction Table

**Noise adaptive color correction table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_EGP_1 | 0x0250 | 0x04 | 0x50 | 0x05dc | EGP = 6000 x 4/16 = 1500 = 0x05dc |
| NACC_SAT_1 | 0x0252 | 0x04 | 0x52 | 0x0000 | Saturation = 0.0, 0% |
| NACC_EGP_2 | 0x0254 | 0x04 | 0x54 | 0x0465 | EGP = 6000 x 3/16 = 1150 = 0x0465 |
| NACC_SAT_2 | 0x0256 | 0x04 | 0x56 | 0x0040 | Saturation = 0.25, 25% |
| NACC_EGP_3 | 0x0258 | 0x04 | 0x58 | 0x02ee | EGP = 6000 x 2/16 = 750 = 0x02ee |
| NACC_SAT_3 | 0x025a | 0x04 | 0x5a | 0x0080 | Saturation = 0.5, 50% |
| NACC_EGP_4 | 0x025c | 0x04 | 0x5c | 0x0177 | EGP = 6000 x 1/16 = 375 = 0x0177 |
| NACC_SAT_4 | 0x025e | 0x04 | 0x5e | 0x00c0 | Saturation = 0.75, 75% |
| NACC_EGP_5 | 0x0260 | 0x04 | 0x60 | 0x0000 | EGP = 0 = 00 = 0x0000 |
| NACC_SAT_5 | 0x0262 | 0x04 | 0x62 | 0x0100 | Saturation = 1.0, 100% |
| NACC_EGP_6 | 0x0264 | 0x04 | 0x64 | 0x0000 | |
| NACC_SAT_6 | 0x0266 | 0x04 | 0x66 | 0x0000 | |
| NACC_EGP_7 | 0x0268 | 0x04 | 0x68 | 0x0000 | |
| NACC_SAT_7 | 0x026a | 0x04 | 0x6a | 0x0000 | |
| NACC_EGP_8 | 0x026c | 0x04 | 0x6c | 0x0000 | |
| NACC_SAT_8 | 0x026e | 0x04 | 0x6e | 0x0000 | |

## Bright Coefficients Table

**Noise adaptive color correction bright coefficients table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_BC_00 | 0x0270 | 0x04 | 0x70 | 0x0235 | Value for CC_COEF_00 |
| NACC_BC_01 | 0x0272 | 0x04 | 0x72 | 0xff46 | Value for CC_COEF_01 |
| NACC_BC_02 | 0x0274 | 0x04 | 0x74 | 0xff85 | Value for CC_COEF_02 |
| NACC_BC_10 | 0x0276 | 0x04 | 0x76 | 0xff64 | Value for CC_COEF_10 |
| NACC_BC_11 | 0x0278 | 0x04 | 0x78 | 0x01fc | Value for CC_COEF_11 |
| NACC_BC_12 | 0x027a | 0x04 | 0x7a | 0xff9f | Value for CC_COEF_12 |
| NACC_BC_20 | 0x027c | 0x04 | 0x7c | 0x0008 | Value for CC_COEF_20 |
| NACC_BC_21 | 0x027e | 0x04 | 0x7e | 0xfe8d | Value for CC_COEF_21 |
| NACC_BC_22 | 0x0280 | 0x05 | 0x00 | 0x026b | Value for CC_COEF_22 |

## Dark Coefficients Table

**Noise adaptive color correction dark coefficients table**

| Name | Address | Block | Offset | Value | Description |
|---|---|---|---|---|---|
| NACC_DC_00 | 0x0282 | 0x05 | 0x02 | 0x0048 | Value for CC_COEF_00 |
| NACC_DC_01 | 0x0284 | 0x05 | 0x04 | 0x010b | Value for CC_COEF_01 |
| NACC_DC_02 | 0x0286 | 0x05 | 0x06 | 0xffaa | Value for CC_COEF_02 |
| NACC_DC_10 | 0x0288 | 0x05 | 0x08 | 0x0048 | Value for CC_COEF_10 |
| NACC_DC_11 | 0x028a | 0x05 | 0x0a | 0x010b | Value for CC_COEF_11 |
| NACC_DC_12 | 0x028c | 0x05 | 0x0c | 0xffaa | Value for CC_COEF_12 |
| NACC_DC_20 | 0x028e | 0x05 | 0x0e | 0x0048 | Value for CC_COEF_20 |
| NACC_DC_21 | 0x0290 | 0x05 | 0x10 | 0x010b | Value for CC_COEF_21 |
| NACC_DC_22 | 0x0292 | 0x05 | 0x12 | 0xffaa | Value for CC_COEF_22 |

## RESERVED – Reserved

Address: 0x0294 ⋮ 0x07fe  Block: 0x05 ⋮ 0x0f  Offset: 0x44 ⋮ 0x7e

Access: Read / Write  Reset value: –  Type: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | – | Reserved – DO NOT write any data to these registers |

# 10 Expert Sensor Registers

The image sensor registers are normally controlled by the image pipeline. The registers normally written to are the row and column registers and the configuration registers. If the exposure or gain registers are written to and the auto exposure/white balance is enabled, the new values will be immediately overwritten.

**NOTE** It is possible that by modifying these registers, the sensor will be programmed into a non-functional state. If this occurs reset the module.

## Register Description Notes

The register descriptions show the format of the data in each register. Each description starts with the mnemonic name for the register, followed by the full register name. The 8-bit address is also listed.

The 8 bits of the register are shown, with mnemonics for each bit. Whenever a bit is reserved, the reserved value is shown and the block is grayed out. Below this is a table of details of each bit, with the default value listed in bold.

The access, either read, read/write or mixed is shown, along with the reset value. All registers are 8-bit

### Reserved Registers

**NOTE** If a register is listed as reserved, **DO NOT** write any data to the register. It is possible to program the sensor into a non-functional state. If this occurs reset the module.

**Sensor register index**



**Pixel timing register index**

# Sensor Registers

## IDENT – Identification

Address: 0x0800 Block: 0x10 Offset: 0x00

Access: Read / Write Reset value: 0x60 Type: One 5-bit integer, one 3-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TYPE | | | | | REV | | |

**Identification**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | REV | 000 | Revision:<br>000:Revision A |
| 3 - 7 | TYPE | 01011 | Sensor type:<br>01100: 640 x 480 landscape VGA |

## IS_STATUS – Image Sensor Status

Address: 0x0801 Block: 0x10 Offset: 0x01

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | SSF | SFS | EFS | CC | FC | RC | RF |

**Image sensor status**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | RF | 0 | Run flag - READ ONLY:<br>**0: No image capture process is executing**<br>1: Image capture in progress |
| 1 | RC | 0 | Row complete flag – clear by writing a “1”:<br>**0: Row has NOT been completed since RC flag last cleared**<br>1: Row has been completed since RC flag last cleared |
| 2 | FC | 0 | Frame complete flag – clear by writing a “1”:<br>**0: Frame has NOT been completed since FC flag last cleared**<br>1: Frame has been completed since FC flag last cleared |
| 3 | CC | 0 | Image capture complete flag – clear by writing a “1”:<br>**0: Image capture process has NOT been completed since CC flag last cleared**<br>1: Image capture process has been completed since CC flag last cleared |
| 4 | EFS | 0 | Exposure frame registers sampled – clear by writing a “1”:<br>**0: Shadow registers associated with exposure have NOT been updated with user-accessible contents**<br>1: Shadow registers associated with exposure have been updated with user-accessible contents |
| 5 | SFS | 0 | Sample frame registers sampled – clear by writing a “1”:<br>**0: Shadow registers associated with sample frames have NOT been updated with user-accessible contents**<br>1: Shadow registers associated with sample frames have been updated with user-accessible contents |
| 6 | SSF | 0 | Shutter synchronization flag – clear by writing a “1”:<br>**0: Shutter synchronization has not been detected since the flag was last cleared**<br>1: Shutter is synchronized now |
| 7 | Reserved | 0 | Reserved – hard wired to 0 |

## RESERVED – Reserved

| | | | | | |
|---|---|---|---|---|---|
| Address: | 0x0802<br>0x0803<br>0x0804 | Block: | 0x10 | Offset: | 0x02<br>0x03<br>0x04 |

Access: Read  Reset value: –  Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## ICTRL – Interface Control

Address: 0x0805 Block: 0x10 Offset: 0x05

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | AAD |

**Interface control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | AAD | 0 | Auto address disable:<br>**0: Register address automatically incremented after a register read or write**<br>1: Register address must be set before each read or write |
| 1 - 7 | Reserved | 0000000 | Reserved – can read and write a 1, but bits are unconnected |

## RESERVED – Reserved

| | | | | | |
|---|---|---|---|---|---|
| Address: | 0x0806<br>0x0807<br>0x0808 | Block: | 0x10 | Offset: | 0x06<br>0x07<br>0x08 |
| Access: Read | | Reset value: – | | Type: – | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## ADC_CTRL – ADC Control

Address: 0x0809 Block: 0x10 Offset: 0x09

Access: Read / Write Reset value: 0x01 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | DBS | |

**ADC control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 1 | DBS | 01 | ADC data output:<br>00: 10-bit data<br>**01: 8-bit data**<br>10: 6-bit data<br>11: 4-bit data |
| 2 - 7 | Reserved | 000000 | Reserved – hard wired to 0 |

## FWROW – Window First Row Address

Address: 0x080a Block: 0x10 Offset: 0x0a

Access: Read / Write Reset value: 0x01 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WFR_ADD | | | | | | | |

**Window first row address**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | WFR_ADD | 00000001 | Window first row address. Represents bits [9:2] of the address of the first row of the image window. Bits [1:0] of the first row address are hard wired to “00” to force the window to begin on an even row boundary that is a multiple of four.<br><br>The legal range is from zero to the window last row address minus three:<br><br>0 <= WFR_ADD[9:0] <= WLR_ADD[9:0] - 3 |

## FWCOL – Window First Column Address

Address: 0x080b Block: 0x10 Offset: 0x0b

Access: Read / Write Reset value: 0x01 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WFC_ADD | | | | | | | |

**Window first column address**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | WFC_ADD | 00000001 | Window first column address. Represents bits [9:2] of the address of the first column of the image window. Bits [1:0] of the first column address are hard wired to “00” to force the window to begin on an even column boundary that is a multiple of four.<br><br>The legal range is from zero to the window last row address minus the minimum number of columns in the image window:<br><br>0 <= WFC_ADD[9:0] <= WLC_ADD[9:0] - MINC + 1<br><br>where MINC is the minimum number of columns in the image window for a given operating mode and column timing. |

## LWROW – Window Last Row Address

Address: 0x080c Block: 0x10 Offset: 0x0c

Access: Read / Write Reset value: 0x7a Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WLR_ADD | | | | | | | |

**Window last row address**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | WLR_ADD | 01111010 | Window last row address. Represents bits [9:2] of the address of the last row of the image window. Bits [1:0] of the last row address are hard wired to “11” to force the window to end on an odd row boundary that is a multiple of four minus 1.<br><br>The legal range is from the address of the first row plus 3 to the number of rows minus one:<br><br>WFR_ADD[9:0] + 3 <= WLR_ADD[9:0] <= number of rows in array - 1 |

## LWCOL – Window Last Column Address

Address: 0x080d Block: 0x10 Offset: 0x0d

Access: Read / Write Reset value: 0xa2 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WLC_ADD | | | | | | | |

**Window last column address**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | WLC_ADD | 10100010 | Window last column address. Represents bits [9:2] of the address of the last column of the image window. Bits [1:0] of the last column address are hard wired to “11” to force the window to end on an odd column boundary that is a multiple of four minus one.<br><br>The legal range is from the first column address plus the minimum number of columns to the number of columns in the array minus one:<br><br>WFC_ADD[9:0] + MINC -1 <= WLC_ADD[9:0] <= number of columns in the array - 1<br><br>where MINC is the minimum number of columns in the image window for a given operating mode and column timing. |

## CLK_PIXEL – Clocks per Pixel

Address: 0x080e Block: 0x10 Offset: 0x0e

Access: Read / Write Reset value: 0x02 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CLK_PIXEL | | | | | | | |

**Clocks per pixel**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | CLK_PIXEL | 00000010 | Number of clocks to output one pixel of data:<br>**Default: 2 clocks per pixel** |

**NOTE** This register is set by firmware.

The image processor clock equals the module clock divided by the number of clocks per pixel:

PCLK = MCLK/CPP

The sensor array clock equals the module clock, divided by the number of clocks per pixel and that quantity is divided by the number of channels:

SCLK = [MCLK/CPP]/#CHN

## EREC_PGA – Even Row, Even Column (Green 1) PGA Gain

Address: 0x080f Block: 0x10 Offset: 0x0f

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EREC | | | | | | | |

**Even row, even column (green 1) PGA gain**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EREC | 00000000 | PGA gain for pixels in even rows and even columns (green 1) |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed. The actual gain value is: (1 + bit 6)*[1 + bit[5:0]*0.15/(1 + 2*bit 7)].

## EROC_PGA – Even Row, Odd Column (Red) PGA Gain

Address: 0x0810 Block: 0x10 Offset: 0x10

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EROC | | | | | | | |

**Even row, odd column (red) PGA gain**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EROC | 00000000 | PGA gain for pixels in even rows and odd columns (red) |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed. The actual gain value is: (1 + bit 6)*[1 + bit[5:0]*0.15/(1 + 2*bit 7)].

## OREC_PGA – Odd Row, Even Column (Blue) PGA Gain

Address: 0x0811 Block: 0x10 Offset: 0x11

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OREC | | | | | | | |

**Odd row, even column (blue) PGA gain**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | OREC | 00000000 | PGA gain for pixels in odd rows and even columns (blue) |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed. The actual gain value is: (1 + bit 6)*[1 + bit[5:0]*0.15/(1 + 2*bit 7)].

## OROC_PGA – Odd Row, Odd Column (Green 2) PGA Gain

Address: 0x0812 Block: 0x10 Offset: 0x12

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OROC | | | | | | | |

**Odd row, odd column (green 2) PGA gain**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | OROC | 00000000 | PGA gain for pixels in odd rows and odd columns (Green 2) |

**NOTE**

The value of this register is controlled by the auto exposure function. This register is shadowed. The actual gain value is: (1 + bit 6)*[1 + bit[5:0]*0.15/(1 + 2*bit 7)].

## ROWEXP_L – Row Exposure Low

Address: 0x0813 Block: 0x10 Offset: 0x13

Access: Read / Write Reset value: 0x54 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| REXPL | | | | | | | |

**Row exposure low**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | REXPL | 01010100 | Row exposure low register. This register is the lower 8 bits of the 15-bit row exposure register:<br><br>REXP[14:0] = REXPH[6:0],REXPL[7:0<br><br>Row exposure defined the number of row processing periods that the image is exposed. See "Row Processing Period" on page 407 for additional information. |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed.

## ROWEXP_H – Row Exposure High

Address: 0x0814 Block: 0x10 Offset: 0x14

Access: Read / Write Reset value: 0x00 Type: 7-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | REXPH | | | | | | |

**Row exposure high**

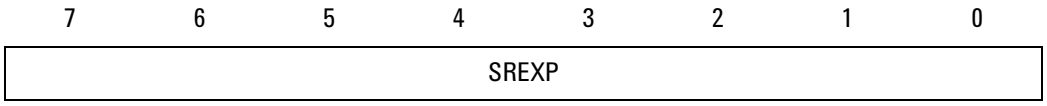
| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | REXPH | 00000000 | Row exposure high register. This register is the upper 7 bits of the 15-bit row exposure register:<br><br>REXP[14:0] = REXPH[6:0],REXPL[7:0]<br><br>Row exposure defined the number of row processing periods that the image is exposed. See "Row Processing Period" on page 407 for additional information. |
| 7 | Reserved | 0 | Reserved – hard wired to 0 |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed.

## SROWEXP – Sub Row Exposure

Address: 0x0815 Block: 0x10 Offset: 0x15

Access: Read / Write Reset value: 0x31 Type: 8-bit integer




**Sub row exposure**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SREXP | 00110001 | Sub row exposure register. Units are column timing periods. Represents bits [9:2] of the sub row exposure count used internally. Bits [1:0] are hard wired to “00” so that the internal sub row exposure value is SUBREXP[9:0={SREXP[7,0],0,0}. The minimum granularity of the sub row exposure is four column timing periods.<br><br>This register is used to specify the sub row duration of time that each pixel is exposed before being sampled. The legal range of values is:<br><br>0 <= SUBREXP[9:0] <= NCTP – MNCT<br><br>(i.e., zero to the number of column timing periods in the image window minus the minimum required number of column timing periods.) |

**NOTE** The value of this register is controlled by the auto exposure function. This register is shadowed.

## ERROR – Error Control

Address: 0x0816 Block: 0x10 Offset: 0x16

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | IIE | IEF | IEE | EEF |

**Error control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | EEF | 0 | Exposure error flag:<br>**0: No error**<br>1: An exposure error was detected since the flag was last cleared. Clear the flag by writing a 0 to the flag and correcting the exposure settings |
| 1 | IEE | 0 | Interrupt when error occurs:<br>**0: No interrupt**<br>1: Interrupt when an exposure error occurs |
| 2 | IEF | 0 | Interface error flag:<br>**0: No error**<br>1: An interface error has occurred since the flag was last cleared. Clear the flag by writing a 0 to the flag. |
| 3 | IIE | 0 | Interrupt when an interface error occurs:<br>**0: No interrupt**<br>1: Interrupt when an interface error occurs |
| 4 - 7 | Reserved | 0000 | Reserved – hard wired to 0 |

## RESERVED – Reserved

Address: 0x0817 0x0818 Block: 0x10 Offset: 0x17 0x18

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## HBLANK – Horizontal Blank

Address: 0x0819 Block: 0x10 Offset: 0x19

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HBLANK | | | | | | | |

**Horizontal blank**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | HBLANK | 000000000 | Horizontal blank.<br>This register represents the number of sensor clock cycles x 2 that are added to the internally controlled minimum horizontal blank period. Legal values are 0 to 255. |

**NOTE** The value of this register is controlled by the simple control register. Depending on the frame rate, clock and window size settings, the value of this register will vary.

## VBLANK – Vertical Blank

Address: 0x081a Block: 0x10 Offset: 0x1a

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VBLANK | | | | | | | |

**Vertical blank**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VBLANK | 000000000 | Vertical blank.<br>This register represents the number of row processing intervals used to either create or increase the vertical blank interval independently of the configured integration time.<br>Legal values are 0 to 255. |

**NOTE** The value of this register is controlled by the simple control register. Depending on the frame rate, clock and window size settings, the value of this register will vary.

## CONFIG_1 – Image Sensor Configuration 1

Address: 0x081b Block: 0x10 Offset: 0x1b

Access: Read / Write Reset value: 0x0e Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | RSS | CSS | CFC | SFC | HPE | MODE |

**Configuration 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | MODE | 0 | Mode – operating mode selection:<br>**0: Normal mode**<br>1: Shutter mode |
| 1 | HPE | 1 | Half power enable:<br>0: Normal operation<br>**1: Low power mode** |
| 2 | SFC | 1 | Stop when frame complete:<br>0: Stop execution immediately on RUN de-assertion. Output data currently in the image pipeline<br>**1: Complete processing of current frame on RUN deassertion** |
| 3 | CFC | 1 | Continuous frame capture:<br>0: Single frame capture when RUN is sent (still mode)<br>**1: Continuous frame capture when RUN is sent video mode)** |
| 4 | CSS | 0 | Column subsample enable – used with CSS4 in CONFIG_2:<br>**0: No subsampling**<br>1: Subsample with a 2:1 ratio |
| 5 | RSS | 0 | Row subsample enable – used with RSS4 in CONFIG_2:<br>**0: No subsampling**<br>1: Subsample with a 2:1 ratio |
| 6 - 7 | Reserved | 00 | Reserved – hard wired to 0 |

## CONTROL_1 – Image Sensor Control 1

Address: 0x081c Block: 0x10 Offset: 0x1c

Access: Read / Write Reset value: 0x24 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | DKMS | PWR | SLCK | ARST | RUN | SLP | RST |

**Image sensor control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | RST | 0 | Hardware reset:<br>**0: Disabled**<br>1: Assert internal hardware reset for sensor block |
| 1 | SLP | 0 | Sleep mode enable:<br>**0: Disabled**<br>1: Sleep mode asserted, internal clocks and state machines disabled, analog blocks disabled |
| 2 | RUN | 1 | Run enable. Transition from 0 to 1 initiates the currently configured image capture process (still or video). Normal termination of a single frame image (still) automatically drives the RUN bit to zero (0).<br><br>When 1, writing a 0 to RUN posts a request to the internal timing controller to stop the current image process. The resulting action is determined by the SFC bit in CONFIG. |
| 3 | ARST | 0 | Global sensor array reset:<br>**0: Disabled**<br>1: Enabled |
| 4 | SLCK | 0 | Shadow register lock:<br>**0: Main register values will be sampled synchronous to internal timing controller cycles**<br>1: Internal shadow registers are locked and will retain their values, modifications to main register settings will be ignored |
| 5 | PWR | 1 | Low power mode:<br>0: Disabled<br>**1: Enabled. When RUN is deasserted, power to all sensor blocks, except this register, is disabled** |

**Image sensor control (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 6 | DKMS | 0 | Dark current measurement mode:<br>**0: Disabled – automatic dark current subtraction is enabled, dark current cannot be measured accurately**<br>1: Enabled – automatic dark current subtraction is disabled, which allows actual dark current to be measured |
| 7 | Reserved | 0 | Reserved – hard wired to 0 |

## RESERVED – Reserved

Address: 0x081d ⋮ 0x0826 Block: 0x10 Offset: 0x1d ⋮ 0x26

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## CONFIG_2 – Image Sensor Configuration 2

Address: 0x0827 Block: 0x10 Offset: 0x27

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | VFLIP | HFLIP | 0 | RSS4 | CSS4 |

**Image sensor configuration 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | CSS4 | 0 | 4:1 Column subsampling – used with CSS in CONFIG_1:<br>**0: See CSS in CONFIG_1**<br>1: Subsample columns with 4:1 ratio, if CSS = 0 |
| 1 | RSS4 | 0 | 4:1 Row subsampling – used with RSS in CONFIG_1:<br>**0: See RSS in CONFIG_1**<br>1: Subsample rows with 4:1 ratio, if RSS = 0 |
| 2 | Reserved | 0 | Reserved – hard wired to 0 |
| 3 | HFLIP | 0 | Horizontal flip control:<br>**0: Image read out from left to right**<br>1: Image read out from right to left |
| 4 | VFLIP | 0 | Vertical flip control:<br>**0: Image read out from bottom to top**<br>1: Image read out from top to bottom |
| 5 - 7 | Reserved | 000 | Reserved – hard wired to 0 |

## GRR_CTRL – Ground Reset Reference Control

Address: 0x0828 Block: 0x10 Offset: 0x28

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | GRR | | RBLS | | | RBE |

**Ground reset reference control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | RBE | 0 | Reset buffer enable:<br>**0: Disabled**<br>1: Enabled |
| 1 - 3 | RBLS | 000 | Reset buffer level select:<br>**000: 2.3 volts**<br>001: 2.4 volts<br>010: 2.5 volts<br>011: 2.6 volts<br>100: 2.7 volts<br>101: 2.8 volts<br>110: 2.9 volts<br>111: 3.0 volts |
| 4 - 5 | GRR | 0 | Ground reset reference – boost cap precharge:<br>**00: Level 0**<br>01: Level 1<br>10: Level 2<br>11: Level 3 |
| 6 - 7 | Reserved | 00 | Reserved |

**NOTE** RBLS must be less than 400 mV above minimum analog $V_{CC}$.

## RESERVED – Reserved

Address: 0x0829 ⋮ 0x0836 Block: 0x10 Offset: 0x29 ⋮ 0x36

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## BIAS_TRM – Bias Current Trim

Address: 0x0837 Block: 0x10 Offset: 0x37

Access: Read / Write Reset value: 0x00 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | BCS | | |

**Bias current trim**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | BCS | 000 | Analog core bias current select:<br>**000: Nominal**<br>001: Decreased 5%<br>010: Decreased 10%<br>011: Decreased 14%<br>100: Increased 30%<br>101: Increased 21%<br>110: Increased 13%<br>111: Increased 7% |
| 3 - 7 | Reserved | 00000 | Reserved |

## RESERVED – Reserved

Address: 0x0838 ⋮ 0x08d6 Block: 0x10 ⋮ 0x11 Offset: 0x38 ⋮ 0x56

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

# Expert Pixel Timing Registers

The expert pixel timing registers are NOT normally written to under normal operation.

**NOTE** It is possible that by modifying these registers, the sensor will be programmed into a non-functional state. If this occurs reset the module.

## SMP_GR_E2 – Sample, Ground Reference Edge 2

Address: 0x08d7 Block: 0x11 Offset: 0x57

Access: Read / Write Reset value: 0x00 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMP_GR_E2 | | | | | | | |

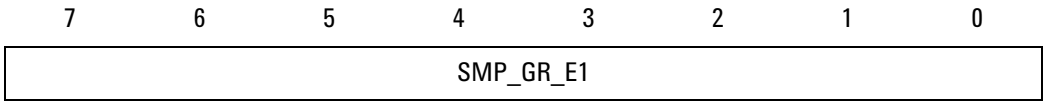
**Sample ground reference edge 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_GR_E2 | 00000000 | Sample ground reference edge 2 |

## SMP_GR_E1 – Sample, Ground Reference Edge 1

Address: 0x08d8 Block: 0x11 Offset: 0x58

Access: Read / Write Reset value: 0x10 Type: 8-bit integer




**Sample ground reference edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_GR_E1 | 00010000 | Sample ground reference edge 1 |

## SMP_GR_E0 – Sample, Ground Reference Edge 0

Address: 0x08d9 Block: 0x11 Offset: 0x59

Access: Read / Write Reset value: 0x0a Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMP_GR_E0 | | | | | | | |

**Sample ground reference edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_GR_E0 | 00001010 | Sample ground reference edge 0 |

## RESERVED – Reserved

Address: 0x08da
0x08db

Block: 0x11

Offset: 0x5a
0x5b

Access: Read

Reset value: –

Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

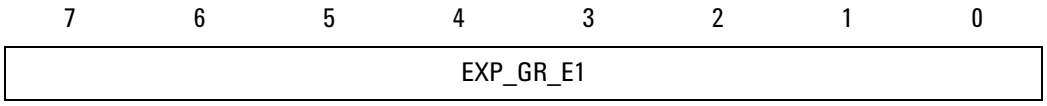
**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## EXP_GR_E1 – Exposure, Ground Reference Edge 1

Address: 0x08dc Block: 0x11 Offset: 0x5c

Access: Read / Write Reset value: 0x10 Type: 8-bit integer



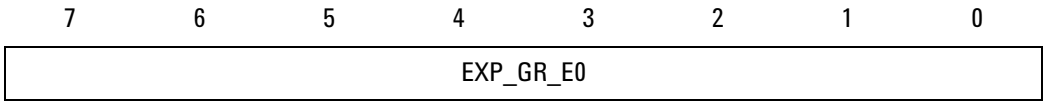

**Exposure ground reference edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_GR_E1 | 00010000 | Exposure ground reference edge 1 |

## EXP_GR_E0 – Exposure, Ground Reference Edge 0

Address: 0x08dd Block: 0x11 Offset: 0x5d

Access: Read / Write Reset value: 0x06 Type: 8-bit integer




**Exposure ground reference edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_GR_E0 | 00000110 | Exposure ground reference edge 0 |

## RESERVED – Reserved

Address: 0x08de Block: 0x11 Offset: 0x5e

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to this register |

## GR_POL – Ground Reference Polarity

Address: 0x08df Block: 0x11 Offset: 0x5f

Access: Read / Write Reset value: 0xd3 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EN_EXP | P_EXP | 0 | P_PSMP | 0 | 0 | EN_SMP | P_SMP |

**Ground reference polarity**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | P_SMP | 1 | Polarity of ground reference at the start of the sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 1 | EN_SMP | 1 | Enables the ground reference during sample time:<br>0: Disabled<br>**1: Enabled** |
| 2 - 3 | Reserved | 00 | Reserved |
| 4 | P_PSMP | 1 | Polarity of ground reference at the start of the pre-sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 5 | Reserved | 0 | Reserved |
| 6 | P_EXP | 1 | Polarity of ground reference at the start of the exposure time:<br>0: 0 = low<br>**1: 0 = high** |
| 7 | EN_EXP | 1 | Enables the ground reference during the exposure time:<br>0: Disabled<br>**1: Enabled** |

## RESERVED – Reserved

Address: 0x08e0 ⋮ 0x08ea  Block: 0x11  Offset: 0x60 ⋮ 0x6a

Access: Read  Reset value: –  Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## SMP_RST_E2 – Sample, Reset Edge 2

Address: 0x08eb Block: 0x11 Offset: 0x6b

Access: Read / Write Reset value: 0x04 Type: 8-bit integer


**Sample reset edge 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_RST_E2 | 00000100 | Sample reset edge 2 |

## SMP_RST_E1 – Sample, Reset Edge 1

Address: 0x08ec Block: 0x11 Offset: 0x6c

Access: Read / Write Reset value: 0x10 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMP_RST_E1 | | | | | | | |

**Sample reset edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_RST_E1 | 00010000 | Sample reset edge 1 |

## SMP_RST_E0 – Sample, Reset Edge 0

Address: 0x08ed Block: 0x11 Offset: 0x6d

Access: Read / Write Reset value: 0x07 Type: 8-bit integer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SMP_RST_E0 | | | | | | | |

**Sample reset edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_RST_E0 | 00000111 | Sample reset edge 0 |

## RESERVED – Reserved

Address: 0x08ee 0x08ef Block: 0x11 Offset: 0x6e 0x6f

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

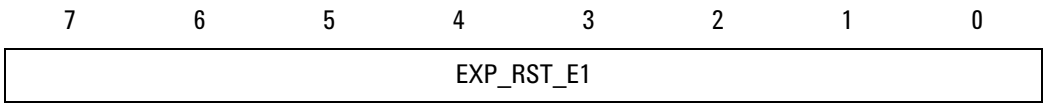
**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

## EXP_RST_E1 – Exposure, Reset Edge 1

Address: 0x08f0 Block: 0x11 Offset: 0x70

Access: Read / Write Reset value: 0x10 Type: 8-bit integer



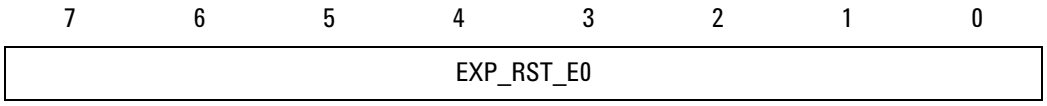

**Exposure reset edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_RST_E1 | 00010000 | Exposure reset edge 1 |

## EXP_RST_E0 – Exposure, Reset Edge 0

Address: 0x08f1 Block: 0x11 Offset: 0x71

Access: Read / Write Reset value: 0x03 Type: 8-bit integer




**Exposure reset edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_RST_E0 | 00000011 | Exposure reset edge 0 |

## RESERVED – Reserved

Address: 0x08f2 Block: 0x11 Offset: 0x72

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to this register |

## RESET_POL – Reset Polarity

Address: 0x08f3 Block: 0x11 Offset: 0x73

Access: Read / Write Reset value: 0xd3 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EN_EXP | P_EXP | 0 | P_PSMP | 0 | 0 | EN_SMP | P_SMP |

**Reset polarity**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | P_SMP | 1 | Polarity of reset at the start of the sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 1 | EN_SMP | 1 | Enables the reset during sample time:<br>0: Disabled<br>**1: Enabled** |
| 2 - 3 | Reserved | 00 | Reserved |
| 4 | P_PSMP | 1 | Polarity of reset at the start of the pre-sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 5 | Reserved | 0 | Reserved |
| 6 | P_EXP | 1 | Polarity of reset at the start of the exposure time:<br>0: 0 = low<br>**1: 0 = high** |
| 7 | EN_EXP | 1 | Enables the reset during the exposure time:<br>0: Disabled<br>**1: Enabled** |

## RESERVED – Reserved

Address: 0x08f4 Block: 0x11 Offset: 0x74

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

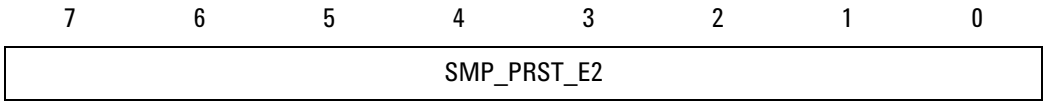
**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to this register |

## SMP_PRST_E2 – Sample, Preset Edge 2

Address: 0x08f5 Block: 0x11 Offset: 0x75

Access: Read / Write Reset value: 0x00 Type: 8-bit integer



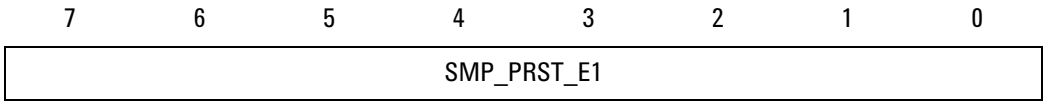

**Sample preset edge 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_PRST_E2 | 00000000 | Sample preset edge 2 |

## SMP_PRST_E1 – Sample, Preset Edge 1

Address: 0x08f6 Block: 0x11 Offset: 0x76

Access: Read / Write Reset value: 0x02 Type: 8-bit integer



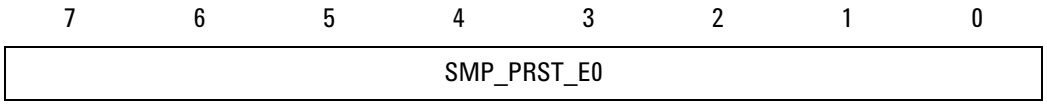

**Sample preset edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_PRST_E1 | 00000010 | Sample preset edge 1 |

## SMP_PRST_E0 – Sample, Preset Edge 0

Address: 0x08f7 Block: 0x11 Offset: 0x77

Access: Read / Write Reset value: 0x0a Type: 8-bit integer




**Sample preset edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SMP_PRST_E0 | 00001010 | Sample preset edge 0 |

## RESERVED – Reserved

Address: 0x08f8 0x08f9 Block: 0x11 Offset: 0x78 0x79

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

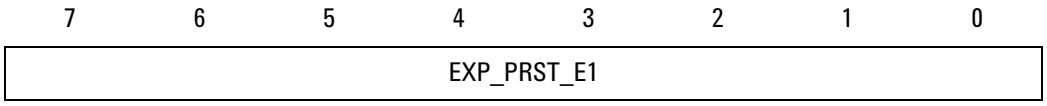
**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to this register |

## EXP_PRST_E1 – Exposure, Preset Edge 1

Address: 0x08fa Block: 0x11 Offset: 0x7a

Access: Read / Write Reset value: 0x02 Type: 8-bit integer



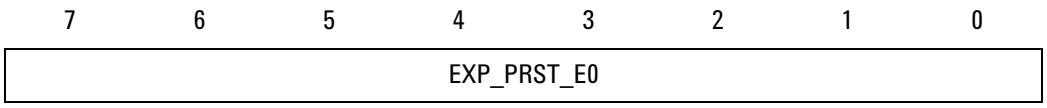

**Exposure preset edge 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_PRST_E1 | 00000010 | Exposure preset edge 1 |

## EXP_PRST_E0 – Exposure, Preset Edge 0

Address: 0x08fb Block: 0x11 Offset: 0x7b

Access: Read / Write Reset value: 0x06 Type: 8-bit integer




**Exposure preset edge 0**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | EXP_PRST_E0 | 00000110 | Exposure preset edge 0 |

## RESERVED – Reserved

Address: 0x08fc Block: 0x11 Offset: 0x7c

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to this register |

## PRESET_POL – Preset Polarity

Address: 0x08fd Block: 0x11 Offset: 0x7d

Access: Read / Write Reset value: 0xd3 Type: Bit field

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EN_EXP | P_EXP | 0 | P_PSMP | 0 | 0 | EN_SMP | P_SMP |

**Preset polarity**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | P_SMP | 1 | Polarity of preset at the start of the sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 1 | EN_SMP | 1 | Enables the preset during sample time:<br>0: Disabled<br>**1: Enabled** |
| 2 - 3 | Reserved | 00 | Reserved |
| 4 | P_PSMP | 1 | Polarity of preset at the start of the pre-sample time:<br>0: 0 = low<br>**1: 0 = high** |
| 5 | Reserved | 0 | Reserved |
| 6 | P_EXP | 1 | Polarity of preset at the start of the exposure time:<br>0: 0 = low<br>**1: 0 = high** |
| 7 | EN_EXP | 1 | Enables the preset during the exposure time:<br>0: Disabled<br>**1: Enabled** |

## RESERVED – Reserved

Address: 0x08fe ⋮ 0x1000 Block: 0x11 ⋮ 0x20 Offset: 0x7e ⋮ 0x00

Access: Read Reset value: – Type: –

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | Reserved | – | Reserved – DO NOT write any data to these registers |

# 11 Expert Image Pipeline Registers

These registers control the details of the operation of the ADCM-2700. They are written to by the camera controller with the values specified by the simple control registers. The registers can be written to directly.

## Register Description Notes

The register descriptions show the format of the data in each register. Each description starts with the mnemonic for the register, followed by the full register name. The full 16-bit address is listed, with the block and offset values that are needed to communicate to the register using the serial control port.

The access, either read/write, read only or mixed is shown, along with the 16-bit reset value (in hex, when the value has a numeric meaning, the decimal equivalent value is shown). The type of register is listed: bit field, real, integer, 2's complement, etc.

The 16 bits of the register are shown, with mnemonics for each bit. Whenever a bit is either unused or reserved, the value for the bit (one or zero) is shown instead of the mnemonic and the block is grayed out.

Next is a table with details for each bit. Shown are the bit location in the register, the mnemonic, the reset value of the bits and the description of all of the valid values of the bit. The default value is shown in bold type.

There are three classes of writable image pipeline registers:

- Class E — Expert register. Can be written to by the user/host at any time and will NOT be changed internally
- Class C — Configuration register. Set internally when the CONFIG bit of the CONTROL register (address 0x0002) is set. These registers may be changed to other values after the CONFIG bit is set, but before setting the SNAP or RUN bits of the CONTROL register
- Class G — Go register. Set internally when the either of the SNAP or RUN bits in the CONTROL register (address 0x0002) are set

**NOTE** Read only registers do not have a class.

**Expert image pipeline register index**

# Main Control Registers

## CMD_1 – Main Command 1

Address: 0x1002 Block: 0x20 Offset: 0x02

Access: Read / Write Reset value: 0x0000 Type: Bit field Class: Mixed

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | DBCG | DAC | SPR | DGO |

**Main command 1**

| Bit(s) | Name | Reset | Description | |
|---|---|---|---|---|
| 0 | DGO | 0 | Data generator "GO" bit:<br>**0: Disabled**<br>1: Enabled | Class G |
| 1 | SPR | 0 | Static pipe reset:<br>**0: Disabled**<br>1: Enabled | Class C |
| 2 | DAC | 0 | Disable auto clear of CMD2:<br>**0: Disabled – auto clear enabled**<br>1: Enabled | Class C |
| 3 | DBCG | 0 | Disable configuration block clock gating:<br>**0: Disabled – configuration block clock gating enabled**<br>1: Enabled – configuration block clock always running | Class C |
| 4 - 15 | Reserved | 0000000000000000 | Reserved | |

## CMD_2 – Main Command 2 (Write 1's only)

Address: 0x1004 Block: 0x20 Offset: 0x04

Access: Read / Write Reset value: 0x0002 Type: Bit field Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | PR | STGO | EOOF |

**Main command 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | EOOF | 0 | Enable output of one frame (give credit):<br>**0: No**<br>1: Yes |
| 1 | STGO | 1 | Statistics go:<br>0: No statistics<br>**1: Gather image statistics** |
| 2 | PR | 0 | Pipe reset:<br>**0: No reset**<br>1: Reset image pipeline |
| 3 - 15 | Reserved | 0000000000000 | Reserved |

**NOTE** Each of the bits, EOOF, STGO and PR autoclear to zero when the function is complete. This register is set by firmware.

## RESERVED – Reserved

Address: 0x1006 Block: 0x20 Offset: 0x06

Access: Read / Write Reset value: – Type: – Class: –

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

**Reserved**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | Reserved | – | Reserved – DO NOT write any data to this register |

## OUTPUT_CTRL – Output Control

Address: 0x1008 Block: 0x20 Offset: 0x08

Access: Read / Write Reset value: 0x9019 Type: Bit field Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| JUST | EOF | SOF | FR_OUT | ESYNC | 4BCCIR | HSMODE | FRVCLK |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P_HYSNC | P_VSYNC | P_VCLK | 1 | OUTPUT_F | | | |

**Output control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 3 | OUTPUT_F | 1001 | Output format:<br>0000: RGB 888 (3 pixels in 3 bytes)<br>0001: RGB 666A tight pack (4 pixels in 9 bytes)<br>0010: RGB 666B loose pack (1 pixel in 3 bytes, left or right justified)<br>0011: RGB 565 (1 pixel in 2 bytes)<br>0100:RGB 444A tight pack (2 pixels in 3 bytes)<br>0101:RGB 444B loose pack (1 pixel in 2 bytes)<br>0110:RGB 444C sparse pack (1 pixel in 3 bytes)<br>0111:RGB 332 (1 pixel in 1 byte)<br>1000:YCbCr 4:2:2 A (Y1, Cb12, Y2, Cr12 order)<br>**1001:YCbCr 4:2:2 B (Cb12, Y1, Cr12, Y2 order)**<br>1010:YCbCr 4:2:2 C (Y1, Cr12, Y2, Cb12 order)<br>1011:YCbCr 4:2:2 D (Cr12, Y1, Cb12, Y2 order)<br>1100:4:4:4 YCbCr<br>1101:4:0:0 grayscale<br>1110:Raw – with AWB and BPA<br>1111:Raw – without AWB and BPA |
| 4 | Reserved | 1 | Reserved – resets to 1 |
| 5 | P_VCLK | 0 | VCLK signal polarity:<br>**0: Data valid on rising edge**<br>1: Data valid on falling edge |
| 6 | P_VSYNC | 0 | VSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |

**Output control (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 7 | P_HSYNC | 0 | HSYNC signal polarity:<br>**0: Valid = high**<br>1: Valid = low |
| 8 | FRVCLK | 0 | Turn off free runing VCLK:<br>**0: VCLK runs all the time the ipipe clock is running**<br>1: VCLK cycles once for each data byte output |
| 9 | HSMODE | 0 | HSYNC/VSYNC mode select:<br>**0: HSYNC high for entire line, VSYNC high for entire frame (HSYNC/VSYNC mode)**<br>1: HSYNC high for last pixel of a line, VSYNC high for the last pixel of a frame. EOL/EOF mode, HSYNC = EOL, VSYNC = EOF (not recommended for use with free running VCLK). |
| 10 | 4BCCIR | 0 | 4-bit CCIR mode:<br>**0: Disabled, CCIR 8-bit mode**<br>1: Enabled, CCIR 4-bit mode, data on D[3:0], D[7:4]=0 |
| 11 | ESYNC | 0 | CCIR embedded synchronization –use ONLY with output formats 8 - 11:<br>MUST set R_Y_MAX_MIN, G_CB_MAX_MIN, B_CR_MAX_MIN:<br>**0: Disabled**<br>1: Enabled |
| 12 | FR_OUT | 1 | Frame output control:<br>0: Frame output controlled by CMD2[0]<br>**1: All frames output** |
| 13 | SOF | 0 | Use start of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 14 | EOF | 0 | Use end of frame codes for each frame:<br>**0: No**<br>1: Yes |
| 15 | JUST | 1 | Output justification – if output pads with zeros:<br>0: Left justified<br>**1: Right justified** |

**NOTE** This register is set by firmware.

## PARALLEL_CTRL – Parallel Output Control

Address: 0x100a Block: 0x20 Offset: 0x0a

Access: Read / Write Reset value: 0x0000 Type: 8-bit integer, Bit field Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | DOLP | 0 | 0 | I_Y/R | I_Cb/G | I_Cr/B | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VCLK_DIV | | | | | | | |

**Parallel output control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | VCLK_DIV | 00000000 | VCLK divider:<br>VCLK = (VCLK_DIV +1) image pipeline clocks per byte |
| 8 | Reserved | 1 | Reserved – resets to 1 |
| 9 | I_Cr/B | 0 | Invert MSB of Cr/B output:<br>**0: No**<br>1: Yes |
| 10 | I_Cb/G | 0 | Invert MSB of Cb/G output:<br>**0: No**<br>1: Yes |
| 11 | I_Y/R | 0 | Invert MSB of Y/R output:<br>**0: No**<br>1: Yes |
| 12 - 13 | Reserved | 00 | Reserved – resets to 0 |
| 14 | DOLP | 0 | Disable output line pacing:<br>**0: Output line pacing enabled**<br>1: Output line pacing disabled |
| 15 | Reserved | 0 | Reserved – resets to 0 |

**NOTE** This register is set by firmware.

## SOF_CODE_W – Working Copy of Start of Frame Code

Address: 0x100c Block: 0x20 Offset: 0x0c

Access: Read / Write Reset value: 0x00ff Type: 8-bit integer Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SOF_CODE_W | | | | | | | |

**Working copy of start of frame code**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | SOF_CODE_W | 11111111 | Start of frame code |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register will be set by firmware. Data will change between still and video modes.

## EOF_CODES_W – Working Copy of End of Frame Codes

Address: 0x100e Block: 0x20 Offset: 0x0e

Access: Read / Write Reset value: 0x0100 Type: Two 8-bit integers Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| BAD_EOF_W | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GOOD_EOF_W | | | | | | | |

**Working copy of end of frame codes**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | GOOD_EOF_W | 00000000 | Good end of frame code |
| 8 - 15 | BAD_EOF_W | 00000001 | Bad end of frame code |

**NOTE** This register will be set by firmware.

## CCIR_TIMING – CCIR Interface Timing

Address: 0x1010 Block: 0x20 Offset: 0x10

Access: Read / Write Reset value: 0x0000 Type: Two 6-bit integers, one 4-bit integer Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| D_HSYNC | | | | V_SETUP | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| V_SETUP | | | | V_HOLD | | | |

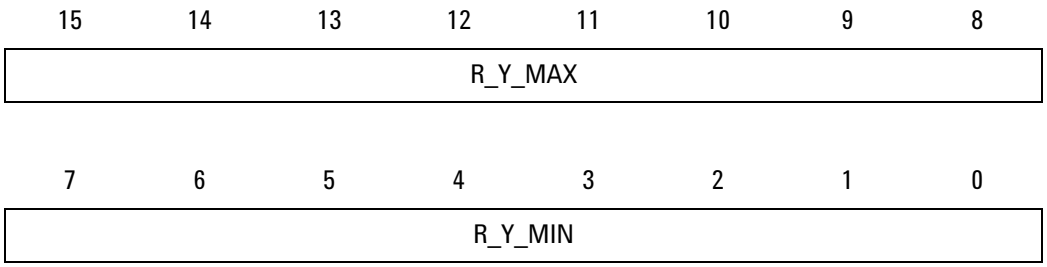
**CCIR interface timing**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 5 | V_HOLD | 000000 | VSYNC hold from HSYNC (in VCLKs) |
| 6 - 11 | V_SETUP | 000000 | VSYNC to HSYNC setup (in VCLKs) |
| 12 - 15 | D_HSYNC | 0000 | Number of dummy HSYNCs |

## R_Y_MAX_MIN – Luminance (or Red) Maximum / Minimum

Address: 0x1012 Block: 0x20 Offset: 0x12

Access: Read / Write Reset value: 0xff00 Type: Two 8-bit integers Class: C



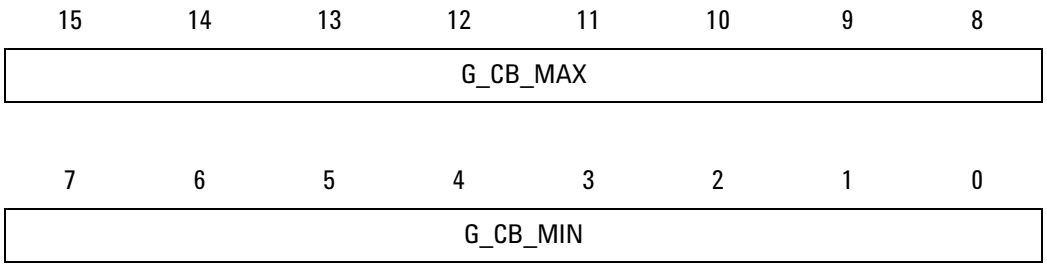

**Luminance (or red) minimum / maximum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | R_Y_MIN | 00000000 | Clip Y or red values to this minimum value |
| 8 - 15 | R_Y_MAX | 11111111 | Clip Y or red values to this maximum value |

## G_CB_MAX_MIN – Chrominance, Cb (or Green) Maximum / Minimum

Address: 0x1014 Block: 0x20 Offset: 0x14

Access: Read / Write Reset value: 0xff00 Type: Two 8-bit integers Class: C




**Chrominance, Cb (or green) minimum / maximum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | G_CB_MIN | 00000000 | Clip Cb or green values to this minimum value |
| 8 - 15 | G_CB_MAX | 11111111 | Clip Cb or green values to this maximum value |

## B_CR_MAX_MIN – Chrominance, Cr (or Blue) Maximum / Minimum

Address: 0x1016 Block: 0x20 Offset: 0x16

Access: Read / Write Reset value: 0xff00 Type: Two 8-bit integers Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| B_CR_MAX | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| B_CR_MIN | | | | | | | |

**Chrominance, Cr (or blue) minimum / maximum**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | B_CR_MIN | 00000000 | Clip Cr or blue values to this minimum value |
| 8 - 15 | B_CR_MAX | 11111111 | Clip Cr or blue values to this maximum value |

## PROCESS_CTRL – Processing Control

Address: 0x1018 Block: 0x20 Offset: 0x18

Access: Read / Write Reset value: 0x0280 Type: Bit field Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| SSA | AVE | V_MIRROR | H_MIRROR | SHARP | | HALF_W | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTW | TMB | CBB | SB | DMB | | G1G2 | BPA |

**Processing control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | BPA | 0 | Bad pixel correction bypass:<br>**0: BPA enabled**<br>1: BPA disabled |
| 1 | G1G2 | 0 | Green pixel filter disable:<br>**0: Filter on**<br>1: Filter off |
| 2 - 3 | DMB | 00 | Demosaic bypass:<br>**00: Use full demosaic**<br>01: Map pixel input to red; zero the green and blue outputs<br>10: Map pixel input to its correct color; zero other two colors<br>11: Superpixel demosaic |
| 4 | SB | 0 | Sizer bypass:<br>**0: Sizer enabled**<br>1: Sizer disabled |
| 5 | CBB | 0 | Color balance bypass:<br>**0: Use color balance matrix**<br>1: No color balance |
| 6 | TMB | 0 | Tonemap bypass:<br>**0: Use tonemap**<br>1: Tonemap disabled |
| 7 | TWW | 1 | Tonemap table weighting:<br>0: Linear spacing<br>**1: Bottom-weighted spacing** |

**Processing control (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 - 9 | HALF_W | 10 | Halftone output width:<br>00: 4-bit<br>01: 6-bit<br>**10: 8-bit – halftone bypassed**<br>11: Reserved (10-bit) |
| 10 - 11 | SHARP | 00 | Sharpening:<br>**00: No effect**<br>01: Low effect<br>10: Medium effect<br>11: High effect |
| 12 | H_MIRROR | 0 | Horizontal mirroring – status only, see note below:<br>**0: Normal output**<br>1: Horizontally mirrored output |
| 13 | V_MIRROR | 0 | Vertical mirroring – status only, see note below:<br>**0: Normal output**<br>1: Vertically mirrored output |
| 14 | AVE | 0 | Anti-vignetting enable:<br>**0: Anti-vignetting disabled**<br>1: Anti-vignetting enabled |
| 15 | SSA | 0 | Sensor 2:1 subsampling active:<br>**0: Tells anti-vignetting that the sensor is not configured for subsampling**<br>1: Tells anti-vignetting that the sensor is configured for 2:1 subsampling |

**NOTE** This register will be set by firmware. Horizontal and vertical mirroring for video mode should be done using the SIZE register (see page 145).

## BPA_SF_GTHRESH – BPA Scale Factor, Green Filter Threshold

Address: 0x101a Block: 0x20 Offset: 0x1a

Access: Read / Write Reset value: 0x0220 Type: One 6-bit integer, one 4-bit integer Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | BPA_SF | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | G_THRESH | | | | | |

**BPA scale factor, green filter threshold**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 5 | G_THRESH | 100000 | Green 1, green 2 filter threshold |
| 6 - 7 | Reserved | 00 | Reserved |
| 8 - 11 | BPA_SF | 0010 | BPA scale factor |
| 12 - 15 | Reserved | 0000 | Reserved |

## BPA_OUTL_PED – BPA Outliers, Pedestal

Address: 0x101c Block: 0x20 Offset: 0x1c

Access: Read / Write Reset value: 0x4008 Type: Two 8-bit integers Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| BPA_OUTL | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BPA_PED | | | | | | | |

**BPA outliers, pedestal**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | BPA_PED | 00001000 | BPA pedestal |
| 8 - 15 | BPA_OUTL | 01000000 | BPA outlier distance |

## BPA_BADPIX_CNT – BPA Bad Pixel Count (Read only)

Address: 0x101e Block: 0x20 Offset: 0x1e

Access: Read only Reset value: 0x0000 Type: 16-bit integer Class: none

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | BAD_PIX | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BAD_PIX | | | | | | | |

**BPA bad pixel count**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 11 | BAD_PIX | 0000 00000000 | Number of “bad” pixels in current frame |
| 12 - 15 | Reserved | 0000 | Reserved |

# Sizer Registers

## SZR_IN_W – Sizer Input Width

Address: 0x1020 Block: 0x20 Offset: 0x20

Access: Read / Write Reset value: 0x0280 Type: 10-bit integer Class: G

Decimal: 640

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | IN_WIDTH | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IN_WIDTH | | | | | | | |

**Sizer input width**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | IN_WIDTH | 10<br>10000000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** This register will be set by firmware.

## SZR_IN_H – Sizer Input Height

Address: 0x1022 Block: 0x20 Offset: 0x22

Access: Read / Write Reset value: 0x01e0 Type: 10-bit integer Class: G

Decimal: 480

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | IN_HEIGHT | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IN_HEIGHT | | | | | | | |

**Sizer input height**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | IN_HEIGHT | 01 11100000 | Number of rows in one frame (without extra Bayer pixels) |
| 10 - 15 | Reserved | 0000000 | Reserved |

**NOTE** This register will be set by firmware.

## SZR_OUT_W – Sizer Output Width

Address: 0x1024 Block: 0x20 Offset: 0x24

Access: Read / Write Reset value: 0x0140 Type: 10-bit integer Class: G

Decimal: 320

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OUT_WID | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_WID | | | | | | | |

**Sizer output width**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | OUT_WID | 01 10110000 | Number of pixels in one row (without extra Bayer pixels) |
| 10 - 15 | Reserved | 0000000 | Reserved |

**NOTE** This register will be set by firmware.

## SZR_OUT_H – Sizer Output Height

Address: 0x1026 Block: 0x20 Offset: 0x26

Access: Read / Write Reset value: 0x00f0 Type: 10-bit integer Class: G

Decimal: 240

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | OUT_HGHT | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OUT_HGHT | | | | | | | |

**Sizer output height**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | OUT_HGHT | 00 11110000 | Number of rows in one frame (without extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** This register will be set by firmware.

# Color Correction

The ADCM-2700 uses a programmable color correction function to adjust for the color filter response of the image sensors. Uncorrected RGB values are offset and then multiplied by a 3 x 3 transform coefficient matrix, the results can the also be offset.

$$\begin{bmatrix} R_{CORRECTED} \\ G_{CORRECTED} \\ B_{CORRECTED} \end{bmatrix} = \left( \begin{bmatrix} CC\_00 & CC\_01 & CC\_02 \\ CC\_10 & CC\_11 & CC\_12 \\ CC\_20 & CC\_21 & CC\_22 \end{bmatrix} \times \left( \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Pre\ Offset\ 0 \\ Pre\ Offset\ 1 \\ Pre\ Offset\ 2 \end{bmatrix} \right) \right) + \begin{bmatrix} Post\ Offset\ 0 \\ Post\ Offset\ 1 \\ Post\ Offset\ 2 \end{bmatrix}$$

## Number Format

The matrix values are stored as a 12-bit signed 2's complement number, with four bits used for the signed integer part of the number and eight bits used for the fractional part of the number.

The numbers can range from -8.000 to +7.996:

- If the number is positive, multiply it by 256 and convert to hex
- If negative, multiply by 256, add to 4096 and convert to hex

These operations will also set the reserved bits to 0. The value of the reserved bits does not matter. Refer to the following table for examples.

**Color correction matrix numbers**

| Matrix Number | x 256 | 4096 + N | Final Hexidecimal |
|---|---|---|---|
| -8.000 | -2048 | 2048 | 0x0800 |
| -7.996 | -2047 | 2049 | 0x0801 |
| – | – | – | – |
| -0.008 | -2 | 4094 | 0x0ffe |
| -0.004 | -1 | 4095 | 0x0fff |
| 0 | 0 | – | 0x0000 |
| 0.004 | 1 | – | 0x0001 |
| – | – | – | – |
| 7.992 | 2046 | – | 0x07fe |
| 7.996 | 2047 | – | 0x07ff |

These registers (0x1028 through 0x1038) are set by the value of bits [1:2] of the ILLUM register (address 0x0010, see page 151) after the CONFIG bit has been set in the CONTROL register (address 0x0002, see page 140). The default value is fluorescent. The following table shows the three set of values.

**Color correction matrix values**

| | D65 (Sunlight) | | Fluorescent | | Tungsten | |
|---|---|---|---|---|---|---|
| **Register** | **Hex** | **Decimal** | **Hex** | **Decimal** | **Hex** | **Decimal** |
| CC_COEF_00 | 0x0235 | 2.20703125 | 0x0235 | 2.20703125 | 0x023a | 2.2265625 |
| CC_COEF_01 | 0x0f8f | – 0.44140625 | 0x0f46 | – 0.7265625 | 0x0f34 | – 0.796875 |
| CC_COEF_02 | 0x0f3b | – 0.76953125 | 0x0f85 | – 0.48046875 | 0x0f92 | – 0.4296875 |
| CC_COEF_10 | 0x0f63 | – 0.61328125 | 0x0f64 | – 0.609375 | 0x0f5a | – 0.6484375 |
| CC_COEF_11 | 0x01f0 | 1.9375 | 0x01fc | 1.984375 | 0x0218 | 2.09375 |
| CC_COEF_12 | 0x0fad | – 0.32421875 | 0x0f9f | – 0.37890625 | 0x0f8e | – 0.4453125 |
| CC_COEF_20 | 0x000d | 0.05078125 | 0x0008 | 0.03125 | 0x0ffa | – 0.0234375 |
| CC_COEF_21 | 0x0eb2 | – 1.3046875 | 0x0e8d | – 1.44921875 | 0x0deb | – 2.08203125 |
| CC_COEF_22 | 0x0241 | 2.25390625 | 0x026b | 2.41796875 | 0x031b | 3.10546875 |

## Color Correction Coefficients

Address: See table Block: 0x20 Offset: See table

Access: Read / Write Reset value: See table Type: 12-bit signed 2's complement (4.8 format) Class: C

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CC_xx | | | |
| Reserved | | | | Signed 2's complement integer | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CC_xx | | | | | | | |
| Fractional part of number | | | | | | | |

**Color correction coefficients registers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | CC_xx | See table | Fractional part of the coefficient. Data in bits 0 - 7 are divided by 256, resulting in the decimal fraction of the coefficient. |
| 8 - 11 | | See table | Signed 2's complement integer part of the coefficient |
| 12 - 15 | Reserved | 0000 | Reserved |

**Color correction coefficients**

| | | | | Reset Value | | Binary | |
|---|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Hex | Decimal | Integer | Fractional |
| CC_COEF_00 | 0x1028 | 0x20 | 0x28 | 0x0235 | 2.20703125 | 0010 | 00110101 |
| CC_COEF_01 | 0x102a | 0x20 | 0x2a | 0x0f46 | – 0.7265625 | 1111 | 01000110 |
| CC_COEF_02 | 0x102c | 0x20 | 0x2c | 0x0f85 | – 0.48046875 | 1111 | 10000101 |
| CC_COEF_10 | 0x102e | 0x20 | 0x2e | 0x0f64 | – 0.609375 | 1111 | 01100100 |
| CC_COEF_11 | 0x1030 | 0x20 | 0x30 | 0x01fc | 1.984375 | 0001 | 11111100 |
| CC_COEF_12 | 0x1032 | 0x20 | 0x32 | 0x0f9f | – 0.37890625 | 1111 | 10011111 |
| CC_COEF_20 | 0x1034 | 0x20 | 0x34 | 0x0008 | 0.03125 | 0000 | 00001000 |
| CC_COEF_21 | 0x1036 | 0x20 | 0x36 | 0x0e8d | – 1.44921875 | 1110 | 10001101 |
| CC_COEF_22 | 0x1038 | 0x20 | 0x38 | 0x026b | 2.41796875 | 0010 | 01101011 |

# Color Correction Offset Registers

## Number Format

The numbers are 2's complement. Since only nine bits are used, the numbers can range from -256 to +255.

If the number is positive, convert it to hex. If negative, add it to 512 and convert it to hex. These operations will also set the reserved bits to 0. The value of the reserved bits does not matter. Refer to the following table for examples.

**Color correction offset numbers**

| Offset Number | N + 512 | Final Hexidecimal |
|---|---|---|
| -256 | 256 | 0x0100 |
| -255 | 257 | 0x0101 |
| – | – | – |
| -2 | 510 | 0x01fe |
| -1 | 511 | 0x01ff |
| 0 | – | 0x0000 |
| 1 | – | 0x0001 |
| – | – | – |
| 254 | – | 0x00fe |
| 255 | – | 0x00ff |

## Color Correction Offsets

Address: See table Block: 0x20 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit signed 2's complement Class: C

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CC_OS_x |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CC_OS_x | | | | | | | |

**Color correction offsets registers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | CC_OS_x | See table | Signed 2's complement value of the offset |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color correction offsets**

| | | | | Reset Value | | |
|---|---|---|---|---|---|---|
| **Name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** | **Binary** |
| CC_PRE_OS_0 | 0x103a | 0x20 | 0x3a | 0x01f8 | – 8 | 111111000 |
| CC_PRE_OS_1 | 0x103c | 0x20 | 0x3c | 0x01f8 | – 8 | 111111000 |
| CC_PRE_OS_2 | 0x103e | 0x20 | 0x3e | 0x01f8 | – 8 | 111111000 |
| CC_POST_OS_0 | 0x1040 | 0x20 | 0x40 | 0x0000 | 0 | 000000000 |
| CC_POST_OS_1 | 0x1042 | 0x20 | 0x42 | 0x0000 | 0 | 000000000 |
| CC_POST_OS_2 | 0x1044 | 0x20 | 0x44 | 0x0000 | 0 | 000000000 |

**NOTE** The values of the pre-offset registers are controlled by auto black level when enabled. Default is for auto black level to be enabled.

# Color Space Conversion – RGB to YUV, RGB to YCbCr

The ADCM-2700 camera module provides a programmable color space conversion function to convert RGB values to other color spaces. RGB values are multiplied by a 3 x 3 transform coefficient matrix and then offset:

$$\begin{bmatrix} \text{New color space 0} \\ \text{New color space 1} \\ \text{New color space 2} \end{bmatrix} = \begin{bmatrix} \text{CSC\_00} & \text{CSC\_01} & \text{CSC\_02} \\ \text{CSC\_10} & \text{CSC\_11} & \text{CSC\_12} \\ \text{CSC\_20} & \text{CSC\_21} & \text{CSC\_22} \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} \text{Offset 0} \\ \text{Offset 1} \\ \text{Offset 2} \end{bmatrix}$$

## RGB to YUV

The default conversion is from RGB to YUV. The color space conversion coefficients for this operation is:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299000 & 0.587000 & 0.114000 \\ -0.147407 & -0.280391 & 0.436798 \\ 0.614777 & -0.514799 & -0.099978 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

However, due to the number format and number of stored bits in the module, the actual matrix used is:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.296875 & 0.585938 & 0.117188 \\ -0.148438 & -0.289063 & 0.437500 \\ 0.617188 & -0.515625 & -0.101563 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

## RGB to YCbCr

If the conversion from RGB to YCbCr is wanted, the default matrix is:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299000 & 0.587000 & 0.114000 \\ -0.168736 & -0.331264 & 0.500000 \\ 0.500000 & -0.418688 & -0.081312 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

Again, due to the number format and the number of bits stored in the module, the actual matrix used is:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.296875 & 0.585938 & 0.117188 \\ -0.171875 & -0.328125 & 0.500000 \\ 0.500000 & -0.421875 & -0.078125 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

**NOTE** The matrix values are copied from addresses 0x0190 to 0x01a6 whenever the RUN bit is set, or addresses 0x01a8 to 0x01be whenever SNAP is set.

## Number Format

The matrix values are stored as a 9-bit signed 2's complement number, with two bits used for the signed integer part of the number and seven bits used for the fractional part of the number.

The numbers can range from - 2.000 to +1.996:

- If the number is positive, multiply it by 128 and convert to hex
- If negative, multiply by 128, add to 512 and convert to hex

These operations will also set the reserved bits to 0. Refer to the following table for examples.

**Color space conversion numbers**

| Matrix Number | x 128 | N + 512 | Final Hexidecimal |
|---|---|---|---|
| -2.000 | -256 | 256 | 0x0100 |
| -1.992 | -255 | 257 | 0x0101 |
| – | – | – | – |
| -0.008 | -1 | 511 | 0x01ff |
| 0 | 0 | | 0x0000 |
| 0.008 | 1 | | 0x0001 |
| – | – | – | – |
| 1.988 | 254 | | 0x00fe |
| 1.996 | 255 | | 0x00ff |

The following table shows the hexidecimal values to write to the registers for both YUV and YCbCr color space conversions.

**Hexidecimal register values for color space conversions**

| "Run" registers | "Snap" registers | YUV | YCbCr |
|---|---|---|---|
| 0x0190 | 0x01a8 | 0x0026 | 0x0026 |
| 0x0192 | 0x01aa | 0x004b | 0x004b |
| 0x0194 | 0x01ac | 0x000f | 0x000f |
| 0x0196 | 0x01ae | 0x01ea | 0x01ed |
| 0x0198 | 0x01b0 | 0x01db | 0x01de |
| 0x019a | 0x01b2 | 0x0038 | 0x0040 |
| 0x019c | 0x01b4 | 0x004f | 0x0040 |
| 0x019e | 0x01b6 | 0x01be | 0x01ca |
| 0x01a0 | 0x01b8 | 0x01f3 | 0x01f6 |

## Color Space Conversion Coefficients

Address: See table Block: 0x20 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit signed 2's complement (2.7 format) Class: G

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_xx |
| Reserved | | | | | | | Integer |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_xx | | | | | | | |
| Integer | Fractional part of number | | | | | | |

**Color space conversion coefficients**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | CSC_xx | See table | Fractional part of the coefficient. Data in bits 0 - 6 are divided by 128, resulting in the decimal fraction of the coefficient. |
| 7 - 8 | | See table | Signed 2's complement integer part of the coefficient |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion coefficients**

| | | | | Reset Value | | Binary | |
|---|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Hex | Decimal | Integer | Fractional |
| CSC_COEF_00 | 0x1046 | 0x20 | 0x46 | 0x0026 | 0.296875 | 00 | 0100110 |
| CSC_COEF_01 | 0x1048 | 0x20 | 0x48 | 0x004b | 0.5859375 | 00 | 1001011 |
| CSC_COEF_02 | 0x104a | 0x20 | 0x4a | 0x000f | 0.1171875 | 00 | 0001111 |
| CSC_COEF_10 | 0x104c | 0x20 | 0x4c | 0x01ed | – 0.1484375 | 11 | 1101101 |
| CSC_COEF_11 | 0x104e | 0x20 | 0x4e | 0x01db | – 0.2890625 | 11 | 1011011 |
| CSC_COEF_12 | 0x1050 | 0x20 | 0x50 | 0x0038 | 0.4375 | 00 | 0111000 |
| CSC_COEF_20 | 0x1052 | 0x20 | 0x52 | 0x004f | 0.6171875 | 00 | 1001111 |
| CSC_COEF_21 | 0x1054 | 0x20 | 0x54 | 0x01be | – 0.515625 | 11 | 0111110 |
| CSC_COEF_22 | 0x1056 | 0x20 | 0x56 | 0x01f3 | – 0.1015625 | 11 | 1110011 |

## Color Space Conversion Offset Registers

The YCbCr and YUV color spaces require that the Cb, Cr and U, V data be offset by +128. For the offsets, the values are copied from addresses 0x01a2 to 0x01a6 whenever the RUN bit is set, or addresses 0x01ba to 0x01be whenever SNAP is set. The number format range is from -256 to 255.

## Color Space Conversion Offsets

Address: See table Block: 0x20 Offset: See table

Access: Read / Write Reset value: See table Type: 9-bit 2's complement Class: G

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | CSC_OS_x |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CSC_OS_x | | | | | | | |

**Color space conversion offsets**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | CSC_OS_x | 000000000 | Integer offset value |
| 9 - 15 | Reserved | 0000000 | Reserved |

**Color space conversion offsets**

| | | | | Reset Value | | |
|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Hex | Decimal | Binary |
| CSC_OS_0 | 0x1058 | 0x20 | 0x58 | 0x0000 | 0 | 000000000 |
| CSC_OS_1 | 0x105a | 0x20 | 0x5a | 0x0080 | 128 | 010000000 |
| CSC_OS_2 | 0x105c | 0x20 | 0x5c | 0x0080 | 128 | 010000000 |

## DATA_GEN – Test Data Generator

Address: 0x105e Block: 0x20 Offset: 0x5e

Access: Read / Write Reset value: 0x0000 Type: Bit field Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MOTION | D_RATE | P_RATE | | | D_PATT | | |

**Test data generator**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 2 | D_PATT | 000 | Data pattern: input to the image pipeline:<br>**000: Normal mode; no data generated, bits 3 - 7 ignored**<br>001: White - motion is fade to black<br>010: Random colors - motion is different patterns<br>011: Sum of coordinates - motion is increasing offsets<br>100: 8-pixel wide border - motion is increasing border width<br>101: Checkerboard, no motion possible<br>110: Reserved<br>111: Color bars, no motion possible |
| 3 - 5 | P_RATE | 000 | Speed of data sent to the image pipeline:<br>**000: 1 pixel per 2 pipe clocks**<br>001: 1 pixel per 4 pipe clocks<br>010: 1 pixel per 8 pipe clocks<br>011: 1 pixel per 16 pipe clocks<br>100: 1 pixel per 32 pipe clocks<br>101: 1 pixel per 64 pipe clocks<br>110: 1 pixel per 128 pipe clocks<br>111: 1 pixel per 256 pipe clocks |
| 6 | D_RATE | 0 | Actual data rate:<br>**0: Maximum possible up to limit specified**<br>1: Exactly as specified; may overflow if unable to support data rate |
| 7 | MOTION | 0 | Motion: adds motion to the image when enabled:<br>**0: No change between frames unless bits 0 - 2 change**<br>1: Add motion between frames (offsetting/translating) |
| 8 - 15 | Reserved | 00000000 | Reserved |

## HSYNC_PER – Horizontal Synchronization Period

Address: 0x1060 Block: 0x20 Offset: 0x60

Access: Read / Write Reset value: 0x0a8b Type: 16-bit integer Class: G

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| HSYNC_PER | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HSYNC_PER | | | | | | | |

**Horizontal synchronization period**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | HSYNC_PER | 00001010 10001011 | Horizontal synchronization period (in image pipeline clocks) |

**NOTE** This register is set by firmware.

# Auto White Balance Registers

The following four registers are used by the auto white balance function to adjust the white point of the scene, assuming “gray world”. These are digital gains, with values from 0 to 7.992.

## Number Format

The matrix values are stored as a 10-bit real number, with three bits used for the integer part of the number and seven bits used for the fractional part of the number. The numbers can range from 0 to 7.992. Refer to the following table for examples.

**Auto white balance gain numbers**

| Number | x 128 | Final Hexidecimal |
|---|---|---|
| 0 | 0 | 0x0000 |
| 0.008 | 1 | 0x0001 |
| — | — | — |
| 7.984 | 1022 | 0x03fe |
| 7.992 | 1023 | 0x03ff |

## Auto White Balance Registers

Address: See table Block: See table Offset: See table

Access: Read /Write Reset value: See table Type: Real (3.7 format) Class: C – if AWB off
G – if AWB on

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | APS_xxx | |
| Reserved | | | | | | Integer | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| APS_xxx | | | | | | | |
| Integer | Fractional part of number | | | | | | |

**AWB gains**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | APS_xxx | 0000000 | Fractional part of the gain; data in bits 0 - 6 are divided by 128, resulting in the decimal part of the gain |
| 7 - 9 | | 001 | Integer part of the gain |
| 10 - 15 | Reserved | 0000000 | Reserved |

**AWB gain registers**

| | | | | Reset value | | Binary | |
|---|---|---|---|---|---|---|---|
| Name | Address | Block | Offset | Reset | Decimal | Integer | Fractional |
| APS_COEF_GRN1 | 0x1062 | 0x20 | 0x62 | 0x0080 | 1.00 | 001 | 0000000 |
| APS_COEF_RED | 0x1064 | 0x20 | 0x64 | 0x0080 | 1.00 | 001 | 0000000 |
| APS_COEF_BLUE | 0x1066 | 0x20 | 0x66 | 0x0080 | 1.00 | 001 | 0000000 |
| APS_COEF_GRN2 | 0x1068 | 0x20 | 0x68 | 0x0080 | 1.00 | 001 | 0000000 |

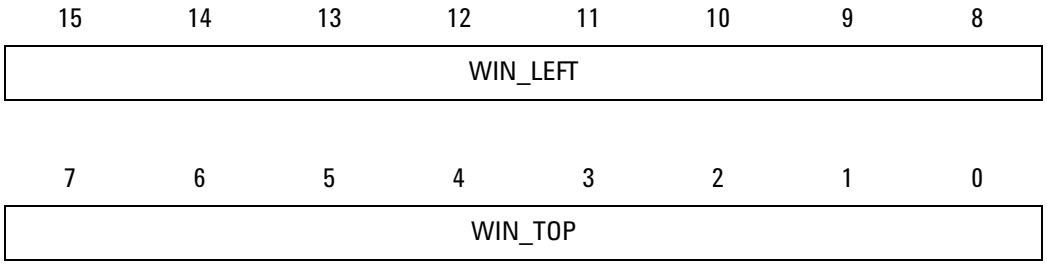
**NOTE** The value of these registers are controlled by firmware if the auto white balance function is enabled.

# Anti-Vignetting Registers

## AV_LEFT_TOP – Anti-Vignetting Window Parameters, Left and Top

Address: 0x106a Block: 0x20 Offset: 0x6a

Access: Read / Write Reset value: 0x0101 Type: Two 8-bit integers Class: C




**Anti-vignetting window parameters, left and top**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | WIN_TOP | 00000001 | Sensor FWROW = 0x01 |
| 9 - 15 | WIN_LEFT | 00000001 | Sensor FWCOL = 0x01 |

**NOTE** This register is set by firmware. If the SIZE register was used, this register will be set automatically. If the window size was manually programmed, then this register must match the current sensor window settings.

## AV_RIGHT_BOT – Anti-Vignetting Window Parameters, Right and Bottom

Address: 0x106c Block: 0x20 Offset: 0x6c

Access: Read / Write Reset value: 0xa27a Type: Two 8-bit integers Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| WIN_RIGHT | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WIN_BOT | | | | | | | |

**Anti-vignetting window parameters, right and bottom**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | WIN_BOT | 01111010 | Set sensor LWROW = 0x7a |
| 9 - 15 | WIN_RIGHT | 10100010 | Set sensor LWCOL = 0xa2 |

**NOTE**

This register is set by firmware. If the SIZE register was used, this register will be set automatically. If the window size was manually programmed, then this register must match the current sensor window settings.

## AV_CENTER_COL – Anti-Vignetting Parameters, Center Column

Address: 0x106e Block: 0x20 Offset: 0x6e

Access: Read / Write Reset value: 0x0148 Type: 10-bit integer Class: C

Decimal: 328

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CENTER_COL | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CENTER_COL | | | | | | | |

**Anti-vignetting window parameters, center column**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 11 | CENTER_COL | 0001 01001000 | Center column of the optical array |
| 12 - 15 | Reserved | 0000 | Reserved |

**NOTE**

This register is set by firmware. If the SIZE register was used, this register will be set automatically. If the window size was manually programmed, then this register must be also manually programmed.

## AV_CENTER_ROW – Anti-Vignetting Parameters, Center Row

Address: 0x1070 Block: 0x20 Offset: 0x70

Access: Read / Write Reset value: 0x00f8 Type: 10-bit integer Class: C

Decimal: 248

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CENTER_ROW | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CENTER_ROW | | | | | | | |

**Anti-vignetting window parameters, center row**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 11 | CENTER_ROW | 0000 111110000 | Center row of the optical array |
| 12 - 15 | Reserved | 0000 | Reserved |

**NOTE** This register is set by firmware. If the SIZE register was used, this register will be set automatically. If the window size was manually programmed, then this register must be also manually programmed.

# Statistics Registers

## STAT_CAP_CTRL – Image Statistics Capture Control

Address: 0x1072 Block: 0x20 Offset: 0x72

Access: Read / Write Reset value: 0x0021 Type: 5-bit integer, bit fields Class: C – if AF on G – if AF off

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LOGSUM | RAMFS | | FCNT | | | | |

**Image statistics capture control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 4 | FCNT | 00001 | Number of frames for statistics; specifies the number of frames used for statistics calculations |
| 5 - 6 | RAMFS | 01 | RAM function selection:<br>00: Off<br>**01: Histogram**<br>10: Reserved<br>11: Flickogram |
| 7 | LOGSUM | 0 | Logarithmic sums:<br>**0: Linear sums**<br>1: Log sums |
| 8 - 15 | Reserved | 00000000 | Reserved |

**NOTE** This register is set by firmware if auto white balance, or auto exposure, or auto flicker is on (The AF, auto functions).

## STAT_MODE_CTRL – Image Statistics Mode Control

Address: 0x1074 Block: 0x20 Offset: 0x74

Access: Read / Write Reset value: 0x0000 Type: 4-bit integer, bit fields Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| FS_CNT | | | | 0 | 0 | BZW | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RZW | | CZW | | LZW | | TZW | |

**Image statistics mode control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 1 | TZW | 00 | Top zone weighting factor for peak and sum image statistics; see below for weighting table |
| 2 - 3 | LZW | 00 | Left zone weighting factor for peak and sum image statistics; see below for weighting table. |
| 4 - 5 | CZW | 00 | Center zone weighting factor for peak and sum image statistics; see below for weighting table. |
| 6 - 7 | RZW | 00 | Right zone weighting factor for peak and sum image statistics; see below for weighting table. |
| 8 - 9 | BZW | 00 | Bottom zone weighting factor for peak and sum image statistics; see below for weighting table. |
| 10 - 11 | Reserved | 00 | Reserved |
| 12 - 15 | FS_CNT | 0000 | Flicker skip count |

**Zone weighting factors**

| Zone Bits | Weighting Factor |
|---|---|
| 00 | 1.00 |
| 01 | 0.50 |
| 10 | 0.25 |
| 11 | 0.00 |

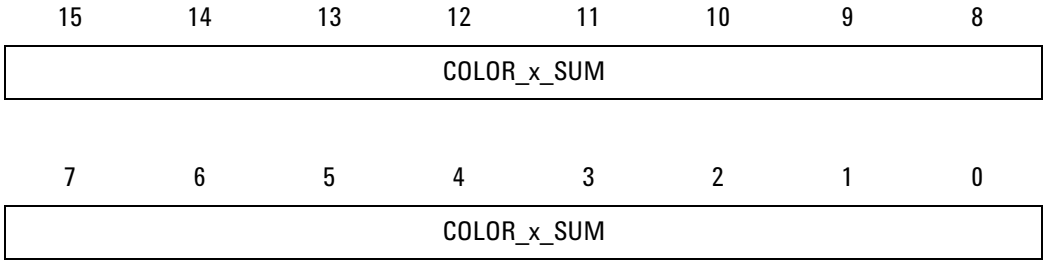
## Pixel Color Sums

The red, green 1, green 2 and blue pixels can be summed into registers.

## Pixel Sum Registers

Address: See table Block: 0x20 Offset: See table

Access: Read only Reset value: See table Type: 16-bit integer Class: none




**Color sum registers**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 15 | COLOR_x_SUM | 00000000 00000000 | Scaled sum of the COLOR_x pixels (sum of all COLOR_x pixels in image)/16384. The sum is accumulated over the 1 to 31 frames as specified by the FCNT field of the STAT_CAP_CTRL register. |

**Color sums**

| Name | Address | Block | Offset | Reset |
|---|---|---|---|---|
| GREEN_1_SUM | 0x1076 | 0x20 | 0x76 | 0x0000 |
| RED_SUM | 0x1078 | 0x20 | 0x78 | 0x0000 |
| BLUE_SUM | 0x107a | 0x20 | 0x7a | 0x0000 |
| GREEN_2_SUM | 0x107c | 0x20 | 0x7c | 0x0000 |

## I_WIDTH – Current Image Width Before Demosaic (Read Only)

Address: 0x107e Block: 0x20 Offset: 0x7e

Access: Read only Reset value: 0x0000 Type: 10-bit integer Class: none

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | I_WIDTH | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| I_WIDTH | | | | | | | |

**Image width**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | I_WIDTH | 00 00000000 | Holds the value of the current image width in pixels + 8 (with extra Bayer pixels) |
| 10 - 15 | Reserved | 000000 | Reserved |

## I_HEIGHT – Current Image Height Before Demosaic (Read Only)

Address: 0x1080 Block: 0x21 Offset: 0x00

Access: Read only Reset value: 0x0000 Type: 10-bit integer Class: none

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | I_HEIGHT | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| I_HEIGHT | | | | | | | |

**Image height**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | I_HEIGHT | 00 00000000 | Holds the value of the current image height in pixels + 8 (with extra Bayer pixels) |
| 10 - 15 | Reserved | 0000000 | Reserved |

# Miscellaneous Registers

## STATUS_FLAGS – Status Flags (Read Only)

Address: 0x1082 Block: 0x21 Offset: 0x02

Access: Read only Reset value: 0x0000 Type: Bit field Class: none

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | APS_BBF | DBBF | OLBE | OLBF | LOF | SRAM_BZY | STAT_BZY |

**Status flags**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | STAT_BZY | 0 | Statistics block busy:<br>0: Not busy<br>**1: Busy** |
| 1 | SRAM_BZY | 0 | Statistics RAM busy:<br>**0: Not busy**<br>1: Busy |
| 2 | LOF | 0 | Last output frame:<br>**0: Bad**<br>1: Good |
| 3 | OLBF | 0 | Output line buffer full:<br>**0: Not full**<br>1: Full |
| 4 | OLBE | 0 | Output line buffer empty:<br>**0: Not empty**<br>1: Empty |
| 5 | DBBF | 0 | Demosaic buffer between frames:<br>**0: Processing frame**<br>1: Between frames |
| 6 | APS_BBF | 0 | APS buffer between frames:<br>**0: APS/BPA processing frame**<br>1: APS/BPA between frames |
| 7 - 15 | Reserved | 000000000 | Reserved |

## CLK_GATE_DIS – Clock Gate Disable

Address: 0x1084 Block: 0x21 Offset: 0x04

Access: Read / Write Reset value: 0x0000 Type: Bit field Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | L_BUF | PACK | CSC | HT | TM | CB |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SIZER | SPIX | I_BLK | I_BUF | BPA | APS | STATS | BIST |

**Clock gate disable**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | BIST | 0 | Bist module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 1 | STATS | 0 | Statistics module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 2 | APS | 0 | APS and anti-vignetting module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 3 | BPA | 0 | BPA module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 4 | I_BUF | 0 | Interpolator buffer module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 5 | I_BLK | 0 | Interpolator module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 6 | SPIX | 0 | Super pixel clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 7 | SIZER | 0 | Sizer module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |

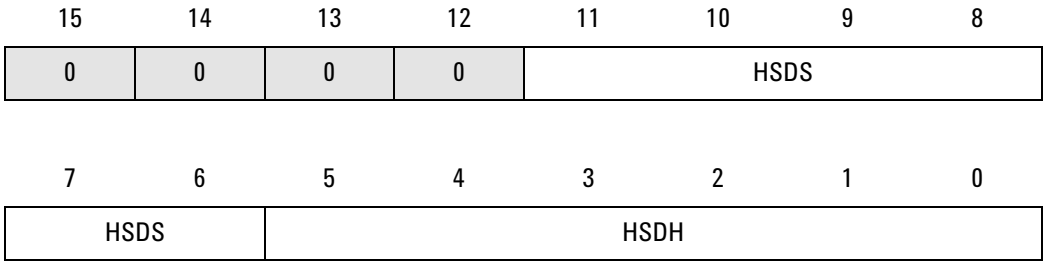
**Clock gate disable (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 | CB | 0 | Color balance module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 9 | TM | 0 | Tonemap module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 10 | HT | 0 | Halftone module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 11 | CSC | 0 | Color space conversion module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 12 | PACK | 0 | Packer module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 13 | L_BUF | 0 | Line buffer module clock gating disable:<br>**0: Gating enabled – clock off when idle**<br>1: Gating disabled – clock always on |
| 14 - 15 | Reserved | 00 | Reserved – resets to 0 |

## CCIR_TIMING2 – CCIR Interface Timing 2

Address: 0x1086 Block: 0x21 Offset: 0x06

Access: Read / Write Reset value: 0x0000 Type: Two 6-bit integers Class: C




**CCIR timing 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 5 | HSDH | 000000 | HSYNC to data hold; units are pixel pairs when in embedded mode, otherwise VCLKs |
| 6 - 11 | HSDS | 000000 | HSYNC to data setup; units are pixel pairs when in embedded mode, otherwise VCLKs |
| 12 - 15 | Reserved | 0000 | Reserved |

## CCIR_TIMING3 – CCIR Interface Timing 3

Address: 0x1088 Block: 0x21 Offset: 0x08

Access: Read / Write Reset value: 0x0010 Type: 8-bit integer, flag Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | DH |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MHSPT | | | | | | | |

**CCIR timing 3**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 7 | MHSPT | 00010000 | Minimum HSYNC passive time. Units are VCLKs |
| 8 | DH | 0 | Disable HSYNC to indicate truncated frames:<br>**0: HSYNC used to indicate truncated frames**<br>1: No truncated frame indication |
| 9 - 15 | Reserved | 0000000 | Reserved – resets to 0 |

## G1G2_DIAG_THRESH – Green 1/Green 2 Diagonal Threshold

Address: 0x108a Block: 0x21 Offset: 0x0a

Access: Read / Write Reset value: 0x0040 Type: 7-bit integer Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | G1G2D_T | | | | | | |

**Green 1 / green 2 diagonal threshold**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 6 | G1G2D_T | 00010000 | Green 1 / green 2 diagonal filter threshold |
| 7 - 15 | Reserved | 000000000 | Reserved – resets to 0 |

## BPA_D2_THRESH – BPA Second Derivative Threshold

Address: 0x108c Block: 0x21 Offset: 0x0c

Access: Read / Write Reset value: 0x0100 Type: 10-bit integer Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | BPA_D2_T | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BPA_D2_T | | | | | | | |

**BPA second derivative threshold**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 9 | BPA_D2_T | 01 00000000 | BPA second derivative threshold |
| 10 - 15 | Reserved | 000000 | Reserved – resets to 0 |

## SERIAL_CTRL – Serial Control

Address: 0x108e Block: 0x21 Offset: 0x0e

Access: Read / Write Reset value: 0x0000 Type: Bit field Class: E

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | V_EOF_M | V_EOF_A |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| H_EOL_M | H_EOL_A | DO | EOL/F_O | EOF_D | EOL_D | SDO | OP |

**Serial control**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 | OP | 0 | Output protocol:<br>**0: Parallel – rest of SERIAL_CRTL is ignored**<br>1: Serial |
| 1 | SDO | 0 | Serial data order selection or 4-bit CCIR data order:<br>**0: MSB first or upper four bits first**<br>1: LSB first or lower four bits first |
| 2 | EOL_D | 0 | End of line:<br>**0: Enabled**<br>1: Disabled |
| 3 | EOF_D | 0 | End of frame:<br>**0: Enabled**<br>1: Disabled |
| 4 | EOL/F_O | 0 | EOL/ EOF order:<br>**0: EOL first**<br>1: EOF first |
| 5 | DO | 0 | EOL/EOF data order:<br>**0: Data first**<br>1: EOL/EOF first |
| 6 | H_EOL_A | 0 | Horizontal synchronization, EOL active:<br>**0: Inactive**<br>1: See bit 7 |
| 7 | H_EOL_M | 0 | Horizontal synchronization, EOL mode:<br>**0: Surrounds words**<br>1: Surrounds lines |

**Serial control (continued)**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 8 | V_EOF_A | 0 | Vertical synchronization, EOL active:<br>**0: Inactive**<br>1: See bit 9 |
| 9 | V_EOF_M | 0 | Vertical synchronization, EOL mode:<br>**0: Surrounds lines**<br>1: Surrounds frames |
| 10 - 15 | Reserved | 000000 | Reserved |

**NOTE** Bit 1 is a dual use bit. In normal serial mode it controls the data order. If the 4-bit CCIR parallel mode is enabled, it selects which nibble is sent first.

## INTP_CTRL_1 – Interpolation Control 1

Address: 0x1090 Block: 0x21 Offset: 0x10

Access: Read / Write Reset value: 0x0188 Type: Two 6-bit integers Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | INTP_PAR3 | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| INTP_PAR3 | | INTP_PAR1 | | | | | |

**Interpolation control 1**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 5 | INTP_PAR1 | 001000 | Threshold offset value |
| 6 - 11 | INTP_PAR3 | 000110 | Threshold value |
| 12 - 15 | Reserved | 0000 | Reserved |

**NOTE** This register is used by the demosaic algorithm. Do not change any values.

## INTP_CTRL_2 – Interpolation Control 2

Address: 0x1092 Block: 0x21 Offset: 0x12

Access: Read / Write Reset value: 0x00c8 Type: One 6-bit integer, one 3-bit integer Class: C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | INTP_PAR2 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| INTP_PAR2 | | SHARP_THRESH | | | | | |

**Interpolation control 2**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 5 | SHARP_THRESH | 001000 | Sharpening threshold: areas with a “non-smoothness” value less than this value will only get “half” sharpened. Not implemented. |
| 6 - 8 | INTP_PAR2 | 011 | Threshold scale factor, $2^{INTP\_PAR2}$ |
| 9 - 15 | Reserved | 0000000 | Reserved |

**NOTE** This register is used by the demosaic algorithm. Do not change any values.

# More Anti-Vignetting Registers

## AV_OVAL_FACT – Anti-Vignetting Oval Factor

Address: 0x1094 Block: 0x21 Offset: 0x14

Access: Read / Write Reset value: 0x0100 Type: Real (1.8 format) Class: E

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | OVAL_FACT |
| Reserved | | | | | | | integer |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OVAL_FACT | | | | | | | |
| Fractional part of number | | | | | | | |

**Anti-vignetting oval factor**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| | | | Oval factor: Modifies the anti-vignetting algorithm:<br>values < 1 result in a tall/skinny oval<br>value = 1 results in a circle<br>values > 1 result in a short/fat oval |
| 0 - 7 | OVAL_FACT | 00000000 | Fractional part of the coefficient. Data in bits 0 - 7 are divided by 256, resulting in the decimal fraction of the coefficient. |
| 8 | | 1 | Integer part of number |
| 9 - 15 | Reserved | 0000000 | Reserved |

## Anti-Vignetting Offsets

### Number Format

The matrix values are stored as a 6-bit signed integer. The numbers can range from -32 to 31. If the number is positive, convert to hex. If the number is negative, add 64 and then convert to hex. Refer to the following table for examples.

**Anti-vignetting offset numbers**

| Number | + 64 | Final He xi decimal |
|---|---|---|
| – 32 | 32 | 0x0020 |
| – 31 | 33 | 0x0021 |
| — | — | — |
| – 2 | 62 | 0x003e |
| – 1 | 63 | 0x003f |
| 0 | — | 0x0000 |
| 1 | — | 0x0001 |
| — | — | — |
| 30 | — | 0x001e |
| 31 | — | 0x001f |

## Anti-Vignetting Offset Registers

Address: See table Block: 0x20 Offset: See table

Access: Read / Write Reset value: See table Type: 6-bit signed integer Class: E

Decimal: See table

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | AV_OS_x |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AV_OS_x | | | | | | | |

**Anti-vignetting offsets**

| Bit(s) | Name | Reset | Description |
|---|---|---|---|
| 0 - 8 | AV_OS_x | 000000 | Integer offset value |
| 9 - 15 | Reserved | 0000000000 | Reserved |

**Anti-vignetting offsets**

| | | | | Reset Value | |
|---|---|---|---|---|---|
| **Name** | **Address** | **Block** | **Offset** | **Hex** | **Decimal** |
| AV_OS_GREEN1 | 0x1096 | 0x21 | 0x16 | 0x0000 | 0 |
| AV_OS_RED | 0x1098 | 0x21 | 0x18 | 0x0000 | 0 |
| AV_OS_BLUE | 0x109a | 0x21 | 0x1a | 0x0000 | 0 |
| AV_OS_GREEN2 | 0x109c | 0x21 | 0x1c | 0x0000 | 0 |

# 12 Sensor Background

# Programming Reference

The ADCM-2700 supports a wide variety of operating modes, window sizes and locations, exposure control settings, timing control settings, analog to digital conversion settings and data output format settings. Operation of the system is primarily controlled by a set of internal registers that must be initialized to the desired settings prior to use.

All register reads and writes occur using a serial communications interface. When an image capture process is running, image data is output from the ADCM-2700 using a parallel or synchronous serial interface.

## Windowing and Panning

The ADCM-2700 supports a very flexible windowing and panning scheme. It supports numerous window sizes, aspect ratios and positions. While the image window can be placed virtually anywhere within the boundaries of the array, there are some restrictions on the size and location. These restrictions are a function of the selected operating mode and the selected subsampling mode (see “Register Description Notes" on page 270 for additional information).

### Addressable Portions of the Array

The sensor array is comprised of a grid of pixels intended to be used for image capture, wrapped by border rows and columns that are addressable, but not intended to be used for normal image capture.

**Addressable portions of the array**

| Window Type | Decimal Size | Decimal Address Range Coordinates (column, row) | Register Settings (hex) | | | |
|---|---|---|---|---|---|---|
| | | | FWROW | FWCOL | LWROW | LWCOL |
| Standard CIF (with demosaic) | 360 x 296 | (4, 4) – (648, 488) | 01 | 01 | 7a | a2 |

The center of the usable image may be offset by changing the suggested register settings, as long as all constraints are met.

### Window Size and Location Restrictions

The primary restrictions on the window location are that it must fit within the addressable array and the starting row and column must align to an address that is a multiple of 4 (forced by the register set).

The primary restriction on the image window size is that the register set interface forces all windows to be a multiple of four rows and columns, regardless of the operating mode and selected subsample mode. When operating in “normal mode”, there is a further restriction on the minimum window width to ensure that the column processing portion of the row processing times of sufficient duration to perform the required exposure reset. When subsample modes are enabled, the window height and width may require being multiples of 4 or 8.

**Window width restriction based on system configuration**

| Operating Mode | Column Subsample Mode | Minimum Window Width (Columns) | Window Width Restrictions |
|---|---|---|---|
| All | Disabled | 16 | Must be a multiple of 4 |
| | 2:1 | 32 | Must be a multiple of 4 |

**Window height restriction based on system configuration**

| Operating Mode | Column Subsample Mode | Minimum Window Height (Rows) | Window Width Restrictions |
|---|---|---|---|
| All | Disabled | 4 | Must be a multiple of 4 |
| | 2:1 | 4 | Must be a multiple of 4 |

## Power Saving Options

The ADCM-2700 offers a reduced power mode, controlled by an external input signal and several programmable reduced power modes.

### Internally Automatic / Programmable Power Saving Options

Low power mode, also called standby mode, is automatically enabled when the chip is idling and no image capturing process is enabled. In this mode, all clocks to the system (except the serial control interface) are disabled. All clocks to the registers are automatically disabled, except during a read or write to the registers.

The clock manager module manages the task of distributing different frequency clocks to different modules in the chip. The analog core gets a clock called SCLK, which is CPP lower than the input clock frequency. The output modules get clocks called PCLK, which is CPP lower than the input clock frequency.

## Programmable Gain Settings

The material used to create color filter patterns, such as the RGB Bayer pattern combined with the frequency response of the photo diodes combine to produce different degrees of sensitivity to red, green and blue light. The ADCM-2700 allows the host system to compensate for these differences, if necessary, by allowing four different gain settings independently programmed using the four PGA gain registers.

The gain values are applied as even row even column, even row odd column, odd row even column and odd row odd column. Since the programmable gain settings modify the resulting signal levels driving the on-chip ADCs, the gain settings should be factored into the exposure control algorithm.

The voltage gain settings are given by:

- Gain = (1 + bit6 x 3) x (1 + bit[5:0] x 0.0625).

## Internal Timing Controller Operation

The timing controller can be initiated by asserting the RUN bit in master mode. Once initiated, the internal timing controller generates all the control signals and executes all of the sequences required to capture one or more frames of image data. To stop the timing controller, deassert the RUN bit.

The ADCM-2700 supports multiple image capture modes, classified as either major or minor. Major modes define the general operation of an image capture and are mutually exclusive in that only one mode can be applied at a time. Minor modes further refine the behavior of the major mode, can be applied to every major mode, affect the major mode operation in a similar manner and are not mutually exclusive.

## Normal Image Capture Mode

The “normal” image capture mode is the default mode. The exposure duration is internally controlled by implementing a rolling electronic shutter that does not require use of an external mechanical shutter or external flash.

Following assertion of the RUN bit in single frame capture mode, the internal timing controller begins sequentially resetting rows in the selected image window to individually start the integration period of each row. To ensure that the integration period of all rows are equivalent, the rows are reset at a rate equal to the rate at which they will subsequently be sampled.

All row operations occur in sequential order and wrap back to the first row in the image window after operating on the last row. When the first row in the image window has been exposed for the programmed duration, the system samples, converts and outputs data for each row in sequential order. If the system is configured to capture a single frame, this process automatically terminates on completion of the first-frame. If it is configured to capture multiple frames, the process continues and is terminated by host interaction.

The following times affect the first-frame and inter-frame delay periods:

- Tvblank = (VBLANK) x (row processing time)
- Texp = (ROWEXP - 1) x (input clock period)
- Tframe = (number of rows in image window) x (row processing time)

Following are the durations for the delays. The timing sequences can be classified into four different groups:

- **Texp > Tframe, Tvblank ≥ Texp**
  The exposure duration is larger than the time it takes to process one frame; the vertical blanking time is larger than or equal to the exposure duration.

- **Texp > Tframe, Tvblank < Texp**
  The exposure duration is larger than the time it takes to process one frame; the vertical blanking time is smaller than the exposure duration.

- **Texp < Tframe, Tvblank ≥ Texp**
  The exposure duration is less than the time it takes to process one frame; the vertical blanking time is larger or equal to the exposure duration.

- **Texp < Tframe, Tvblank < Texp**
  The exposure duration is less than the time it takes to process one frame; the vertical blanking register is less than the exposure duration.

The basic principal in the interaction between exposure settings and frame time are (in order of decreasing priority):

- Exposure duration for each row must be determined by (ROWEXP - 1) x (input clock period)
- First-frame and inter-frame delays are determined by the larger of Tvblank or (Texp – Tframe), where Texp – Tframe equals 0 when Texp < Tframe

When Tvblank > Texp, regardless of the exposure duration against the time it takes to sample a frame, the first-frame and inter-frame delays are determined by the VBLANK register value.

When Tvblank < Texp, the first-frame delay is determined by Texp. When the exposure duration is more than the time it takes to sample a frame, the inter-frame delay is determined by the larger of Tvblank and Texp – Tframe. When the exposure duration is less than the time it takes to sample a frame, Tvblank determines the inter-frame delay.

In the diagram on the following page, columns “exposure reset row” and “sample reset row” refer to the row number that is being reset to start the exposure and the row number that is being sampled. In equation format, the first and last rows in the image window are referred to as “n” and “m” respectively. Values enclosed in square brackets represent a row number based on the assumptions that n = 0, m = 7.

When the inter-frame delay is present, the image data will be output in single frame bursts with a delay between each burst. When the row exposure duration is larger than the time it takes to process one frame, if enabled and configured as the shutter synchronization signal, the nSYNC signal will be asserted at the end of exposure reset period to facilitate operation of an external shutter or flash. When the row exposure duration is less than the time it takes to process one frame, the shutter synchronization signal will not be asserted.

The sample reset occurs during the row sample portion of the row processing time and occurs immediately after the row is sampled (i.e., the end of the integration time for that row). Similarly, the exposure reset essentially marks the start of the integration time for the row that is reset. The column processing time is a function of the number of columns in the image window, the programmed column timing and the column subsampling mode.

## Image Capture Process (Texp > Tframe, Tvblank > Texp)

Tvblank = VBLANK x (row processing time)
Texp = (ROWEXP +1) x (input clock period)
Tframe = (m - n + 1) x (row processing time)

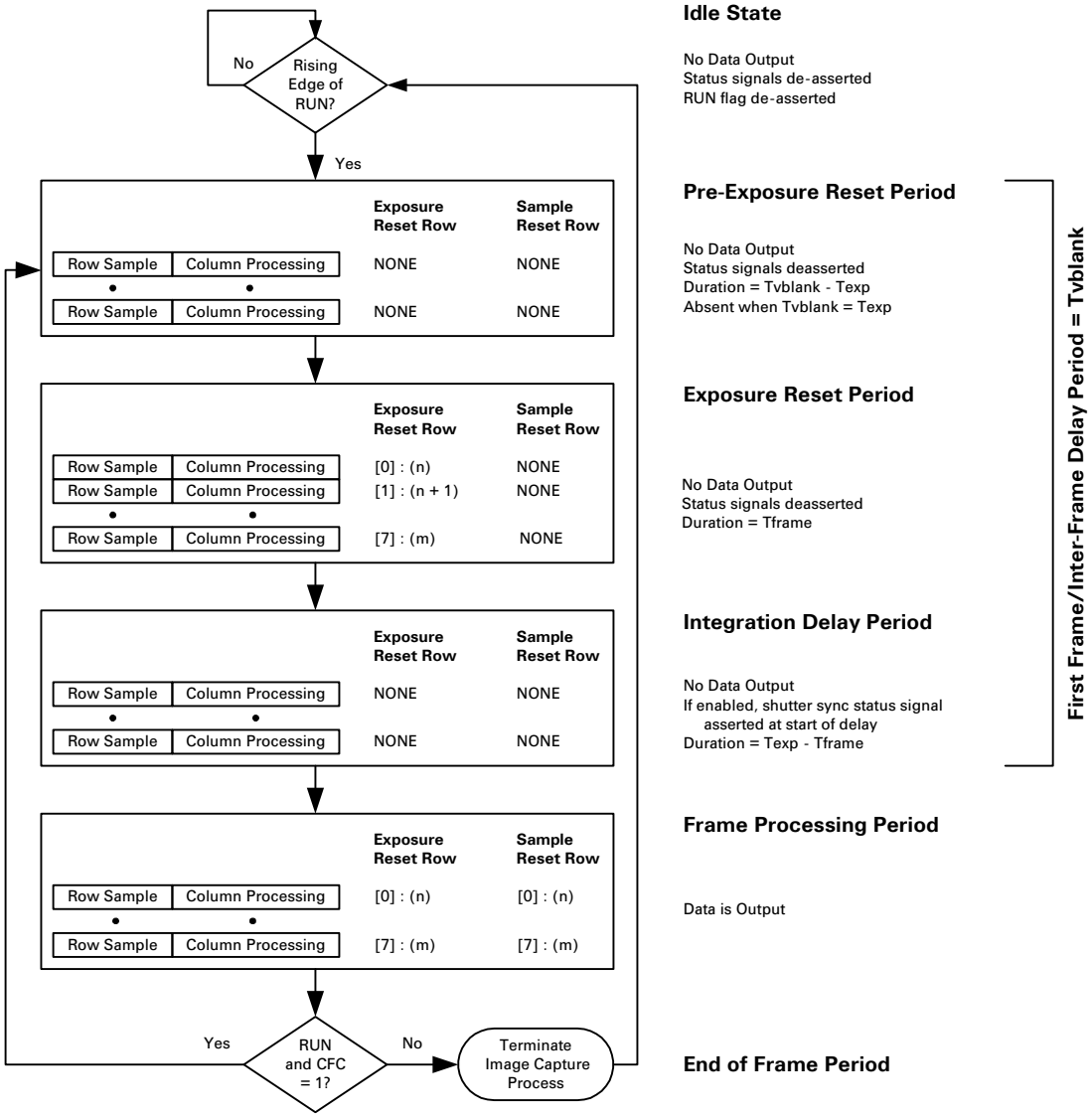



**Figure 90** Image capture process (Texp > Tframe, Tvblank > Texp)

## Image Capture Process (Texp > Tframe, Tvblank > Texp)

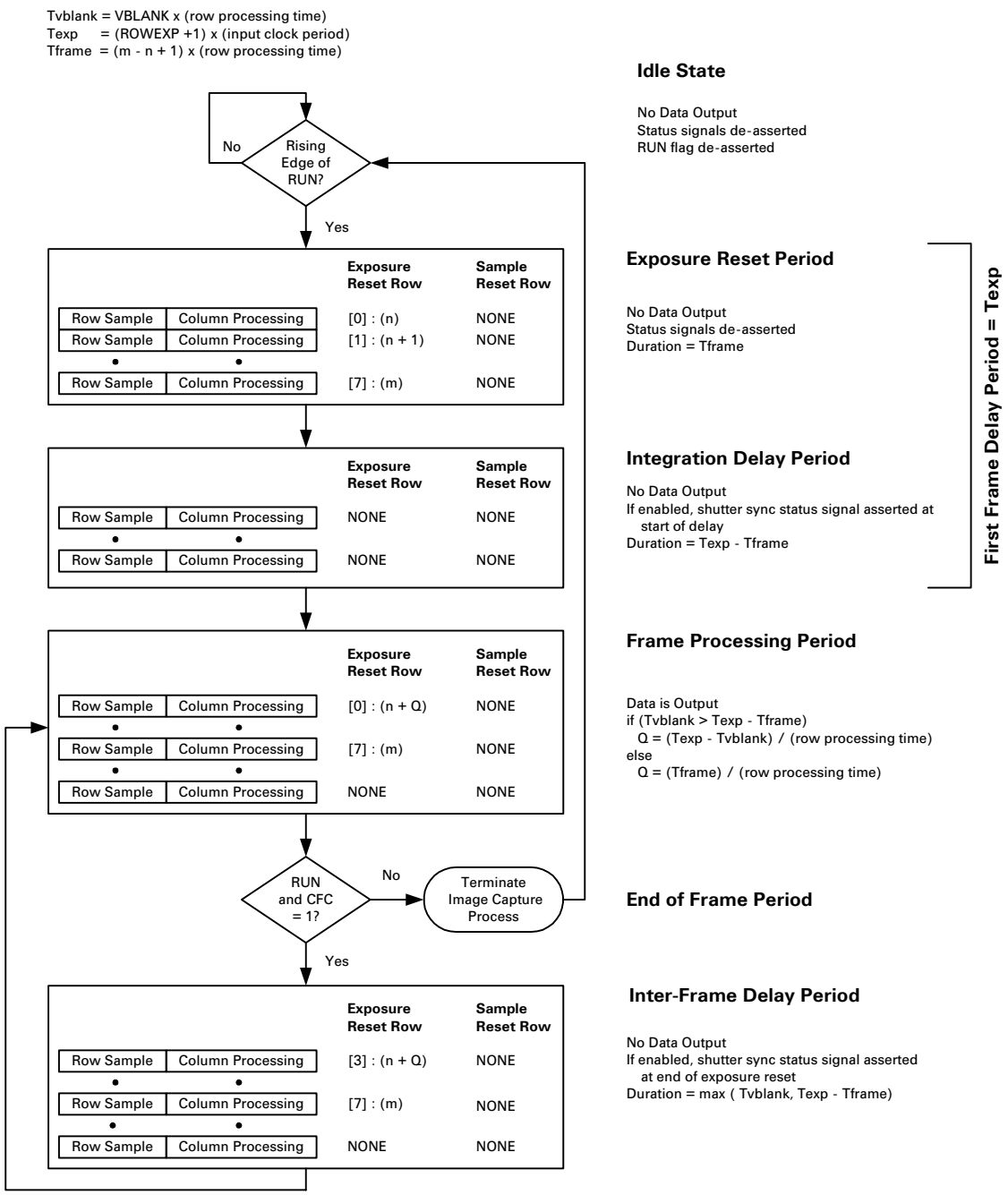



**Figure 91** Image capture process (Texp > Tframe, Tvblank < Texp)

## Image Capture Process (Texp < Tframe, Tvblank > Texp)

The following diagram illustrates the operations sequence for an image capture process with an exposure duration less than the time required to process one frame. The diagram columns “exposure reset row” and “sample reset row” are referring to the row number that is being reset to start the exposure and the row number that is being sampled. In equation format, the first and last rows in the image window are referred to as “n” and “m”, respectively.

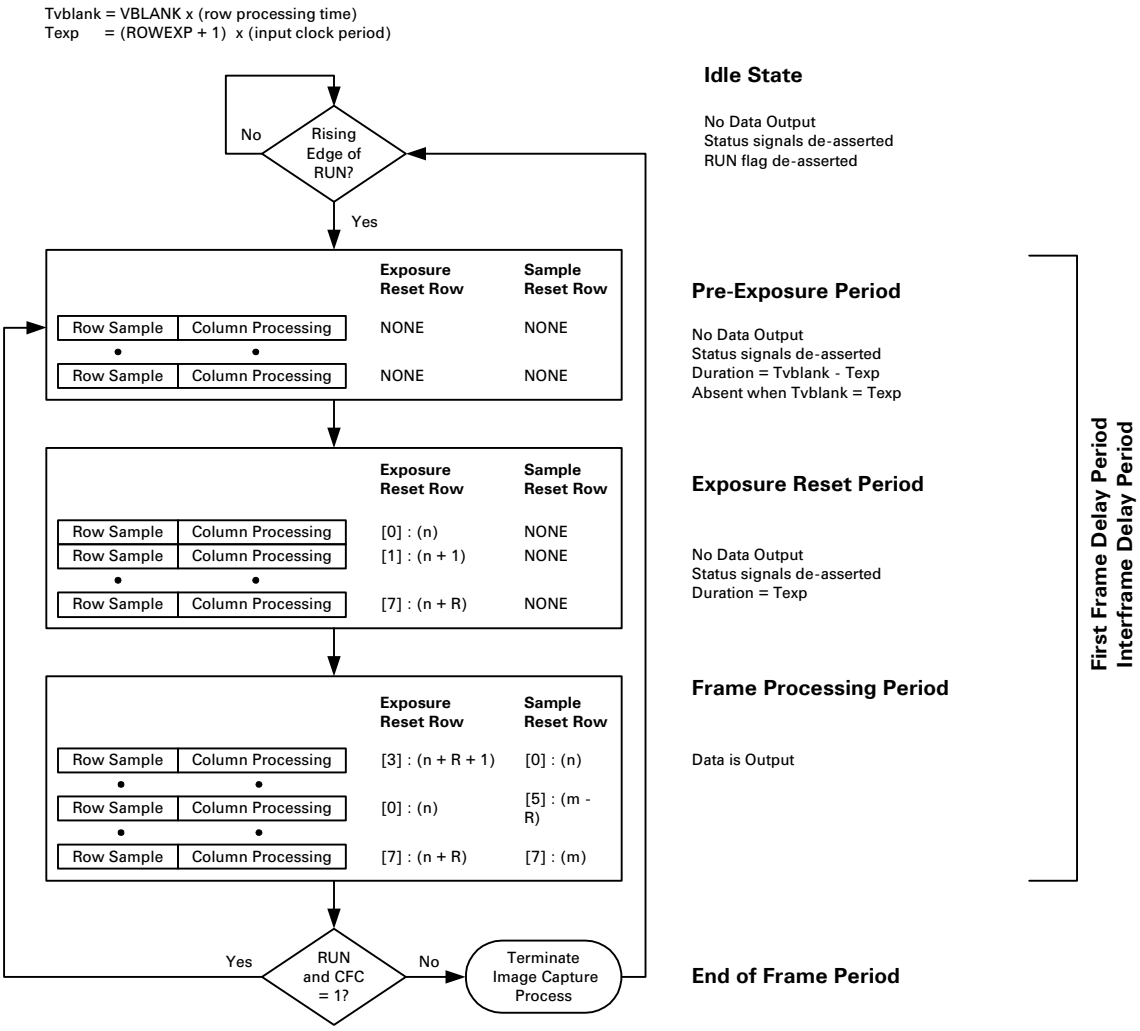



**Figure 92** Image capture process (Texp < Tframe, Tvblank >Texp)

# Image Capture Process (Texp < Tframe, Tvblank < Texp)

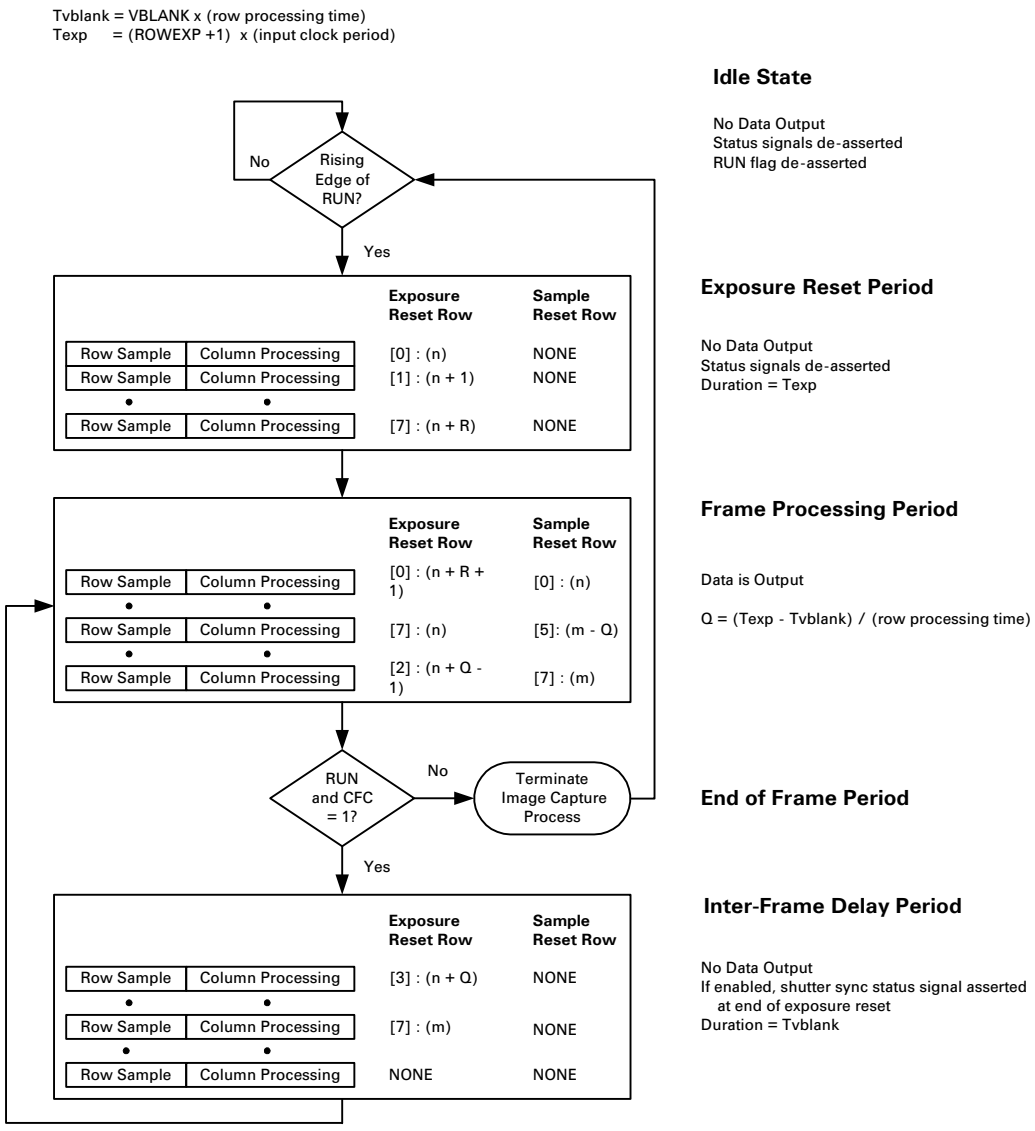
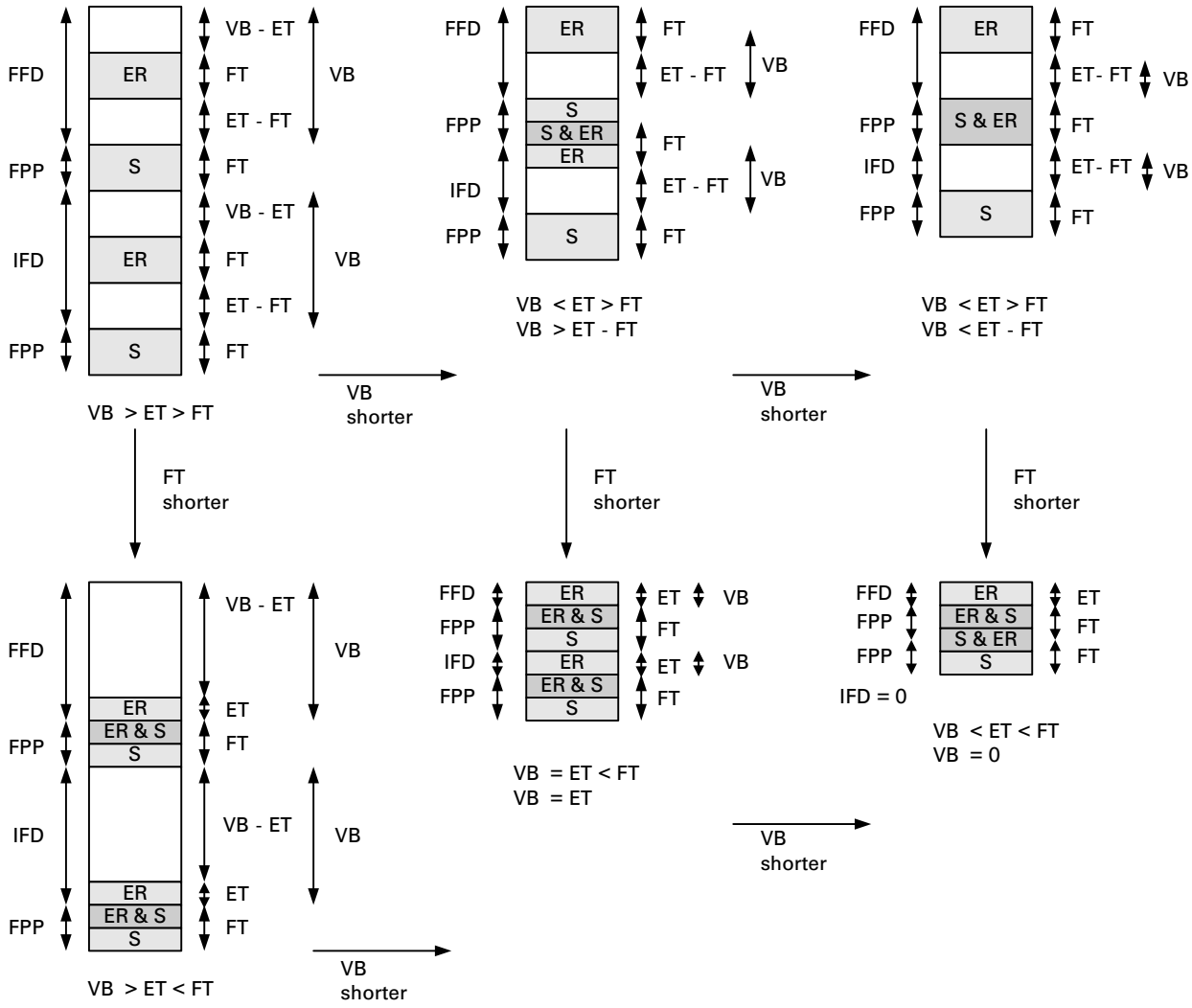



**Figure 93** Image capture process (Texp < Tframe, Tvblank < Texp)

## First-Frame and Inter-Frame Delay Periods

The following diagram provides an overview of the effect of the interaction of VBLANK, the number of rows required for exposure (ET) and the number of rows in the image window, along with the first and inter-frame delay periods.




| | | | |
|---|---|---|---|
| FDD | First Frame Delay | ET | Exposure Time |
| IFD | Inter-Frame Delay | FT | Frame Time |
| FPP | Frame Processing Period | S | Sample |
| VB | Vertical Blank Time | ER | Exposure Reset |

**Figure 94** First-frame and inter-frame delay periods overview effects

## Minor Image Capture Modes

Every minor operating mode can be applied to every major operating mode with a similar effect. Every minor mode can be enabled or disabled, regardless of the state of other minor modes. Supported minor modes include single frame vs. video mode, different channel modes, vertical and horizontal flip modes, and row subsampling and column subsampling.

### Single Frame vs. Video Mode

The states of the SNAP and RUN bits of the CONTROL register determines whether the system operates in single frame or video mode. Setting the RUN bit initiates a video stream of images. SNAP outputs one still frame.

In single frame mode, only one frame of data will be output before the image capture process is stopped. In video mode, the image capture process will continue indefinitely until the host system intervenes by deasserting the RUN bit or asserting a reset condition.

### Vertical and Horizontal Flip Mode

When the SFLIP_UD or the VFLIP_UD bits in CONTROL are set, the row data is output in reverse sequence, from last window row (LWROW) to first window row (FWROW), instead of the normal first to last sequence. All other operations remain the same. Note that the image processor must be able to compensate for the reverse color filter pattern.

When the SFLIP_LF or the VFLIP_LF bit in CONTROL are set, the column data is output in reverse sequence, from last window column (LWCOL) to first window column (FWCOL), instead of the normal first to last sequence. All other operations remain the same.

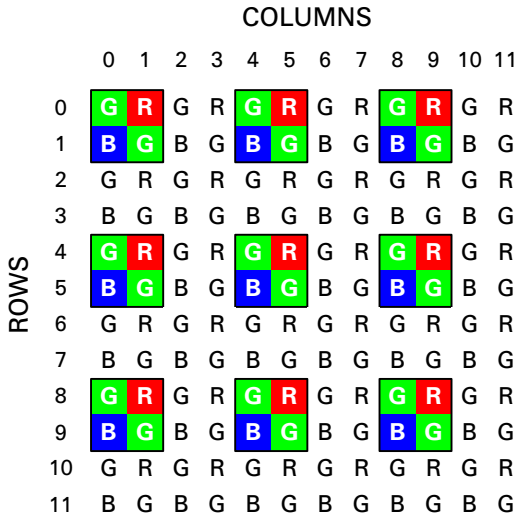
Keep in mind that vertical and horizontal flip modes may be used simultaneously.

**CAUTION** DO NOT set these bits in the image sensor - use the SIZE register. This ensures that the image sensor and the image pipeline are synchronized.

## Subsampling

The row and column subsampling can be used to implement a zoom feature. However, the image quality is not as good as using the sizer function. The subsample modes are controlled by the SSUB and VSUB bits in the SIZE register.

When both the row and column subsampling modes are disabled, each pixel within the defined window is sampled during an image capture process. Enabling subsampling reduces the number of pixels sampled per frame by a factor of four. The impact on the actual frame rate and average data rate also depends on the major operating mode, the exposure registers, the size of the image window and the selected column timing. In general, enabling subsampling increases the frame rate.




2:1 vertical subsampling with take 2, skip 2 pattern
2:1 horizontal subsampling with take 2, skip 2 pattern

**Figure 95** Subsample pattern

# Basic Timing Controller Operations

The internal operation of the timing controller and the rate and timing of data output is deterministic and can be calculated using an understanding of the operating mode in effect and the basic sequences of operations it executes, an understanding of the timing of those basic sequences and pertinent register settings.

The basic sequence of operations executed in each major mode is described in “Normal Image Capture Mode" on page 397. The objective of this section is to describe the timing of the basic sequences and the registers that affect them.

## Row Processing Period

The period of time required to process one row is a fundamental piece of information required to correctly program and understand the rate and timing at which data is output from the chip. Rows are processed sequentially, in ascending order, and the data output rate and overall timing controller sequencing corresponds to the rate at which individual rows are processed. The rate at which rows are processed is highly programmable and depends on the image window size, whether column subsampling is enabled and the selected timing control parameters.

Row processing is comprised of four major portions called the row sample period, the horizontal blanking period, the latency period and the column processing period (as shown below). When operating in normal mode, the exposure reset period occurs during the time slots outside the row sample period. The time delay of the exposure reset period from the row sample period is defined by internal circuitry based on the exposure registers.

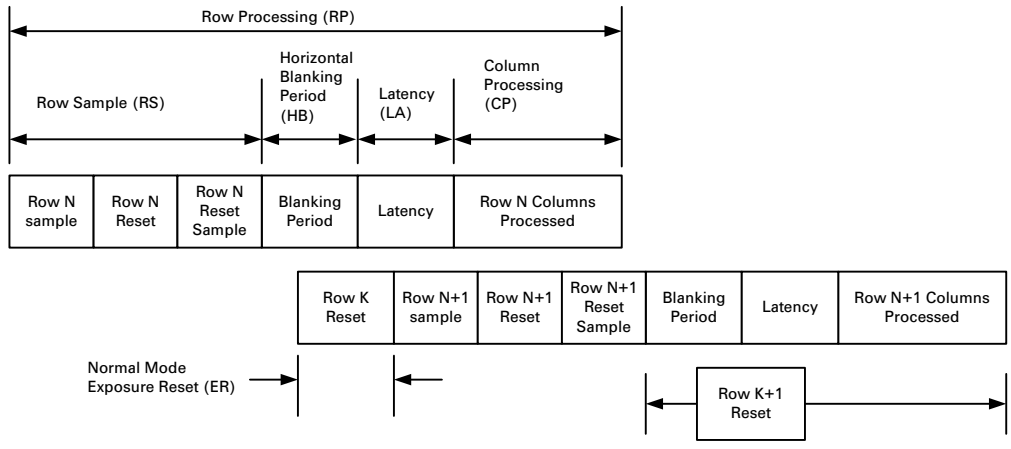



**Figure 96** Row processing period with sub row exposure

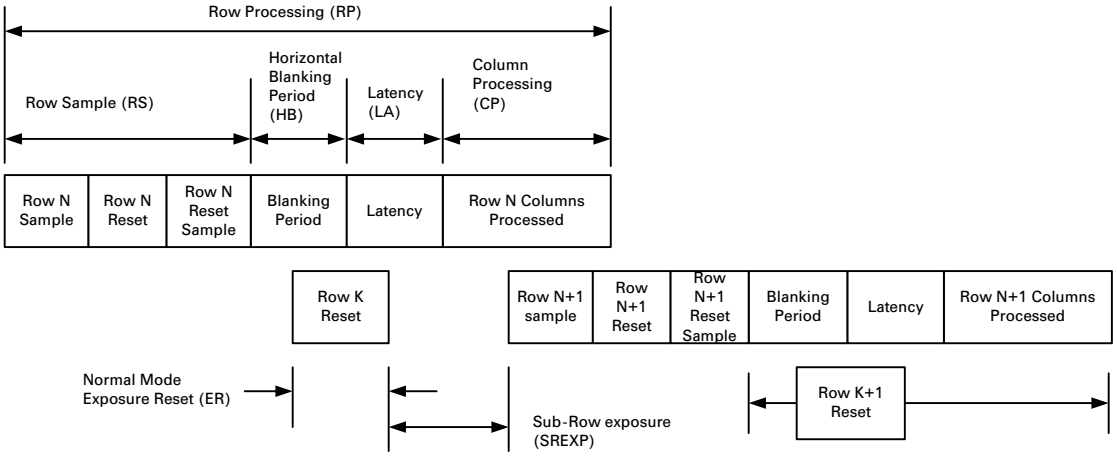



**Figure 97** Row processing period without sub row exposure



### Row Sample Period

The row sample period is independent of the selected operation mode, image window size and the programmed column timing. It is a variable period dependent on the contents of 15 of the expert pixel registers. The following discussion assumes the default values are used. In this case, the row sample (RS) period is 33 MCLK cycles and it is clocked by the sensor master clock, MCLK.

If the current row has been fully exposed, it is sampled, reset and sampled again to implement the correlated double sampling feature. The reset that occurs in this time slot is referred to as the sample reset. If the current row has not been fully exposed, the system does not actually sample or reset any rows, but simply delays an amount of time equivalent to the normal row sample period.

### Horizontal Blanking Period

The duration of the horizontal blanking period is set by the HBLANK register. The HBLANK register is programmable from 0 to 255 and the horizontal blanking period is counted in units of two SCLK cycles per HBLANK count. Horizontal blanking can be 0 to 510 SCLK cycles in duration. This period exists to add delay time to each row so that precise frame rates can be achieved.

### Latency Period

The latency period is a fixed period of ten SCLK clock cycles. This is used to fill up the analog pipeline in preparation for the column processing period.

### Column Processing Period

The analog to digital conversion of a row sampled pixel data occurs during the column processing period, with data output from the chip during this period. The number of SCLK cycles required for the column processing period depends on the number of columns in the image window and the column subsampling. The amount of column subsampling is determined by the CSS field, which is comprised of the concatenation of two separate bits: CSS (register 0x082b, bit 4) and CSS4 (register 0x0827, bit 0).

The column processing period is made up of some number of SCLK periods. The number of column timing periods ranges between two and the number of columns in the array and is programmable, based on the image window size and the operating modes. One column is processed in one column timing period. Columns are normally processed from the left to the right of the array. When HFLP is set, columns are processed from the right to the left of the array.

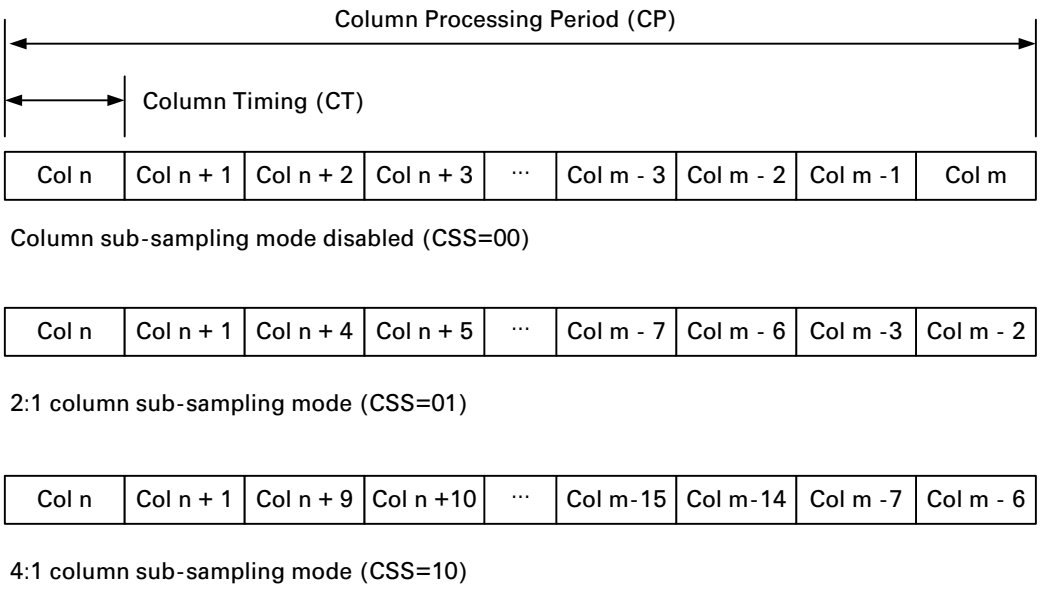



**Figure 98** Column processing sequences

### Frame Processing Period

The frame processing period is the period of time during which one frame of fully exposed data is sampled and output. The frame processing period is highly programmable and depends on the row processing periods, vertical blanking setting and the row subsampling modes.

### First-Frame Delay Period

The first-frame delay period defines the time between when the internal timing controller starts and when data is output. The time depends primarily on the row processing time, the exposure settings and the vertical blanking settings. The first-frame delay period is determined by the larger of Tvblank and Texp. The first-frame delay period can be subdivided into the pre-exposure period, exposure period and the integration delay period. The pre-exposure period is determined by (Tvblank – Texp) and is absent when Texp > Tvblank. The exposure period is determined by the number of rows in the image window and the row processing time. The integration delay period is determined by (Texp – Tframe) and is absent when Tframe > Texp.

### Inter-Frame Delay Period

The inter-frame delay period occurs between two frame processing periods. When Tvblank > Texp, the delay period is determined by Tvblank and is identical to the first-frame delay period. When Tvblank < Texp and Texp < Tframe, the delay is determined by Tvblank. When Tvblank < Texp and Texp > Tframe, the delay is determined by the larger of Tvblank and Texp.

### Fast Rolling Reset Period

The fast rolling reset (FRR) period occurs only in shutter mode. During this period, the internal timing controller will generate an exposure reset cycle for each row of the image window size. The FRR period is equal to the exposure reset period plus FRR overhead times and the number of rows in the image window plus overhead.

$T_{FRR} = N_{ROW} \times (T_{EXP} + T_{FRREXPOHD}) + T_{FRROHD}$

where $T_{EXP} = ER$, $T_{FRREXPOHD} = 1$ and $T_{FRROHD} = 1$

# Timing Equations

**General timing equations**

| Symbol | Equation | Units | Description |
|---|---|---|---|
| $T_{MCLK}$ | | Time | Input system clock (MCLK) period |
| $T_{SCLK}$ | CPP x $T_{MCLK}$ | Time | Analog clock (SCLK) period |
| $T_{PCLK}$ | CPP x $T_{MCLK}$ | Time | Pixel clock (PCLK) period |
| ER | ER=24 | $T_{MCLK}$ | Exposure reset duration |

**Row timing related equations**

| Symbol | Equation | Units | Description |
|---|---|---|---|
| RSS | {rowSS4, rowSS} | | Row subsampling ratio |
| RSf | RSf=29 | $T_{MCLK}$ | Row sample actual time |
| $RS_{SYNC}$ | $RS_{SYNC}$ = 4 | $T_{MCLK}$ | Row sample/column processing overhead |
| RSn | RoundUp((RSf + $RS_{SYNC}$)/TSCLK) | $T_{SCLK}$ | Row sample rounded up to $T_{SCLK}$ period |
| RS | RS=RSn | $T_{SCLK}$ | Row sample period |
| RP | RP=(RS+HB+LA+CP) | $T_{SCLK}$ | Row processing period is determined by the sum of the row sample, horizontal blanking, latency and column processing periods |
| RSSF | RSSF=1 if RSS=00<br>RSSF=2 if RSS=X1<br>RSSF=4 if RSS=10 | | Row subsample scale factor |
| NRP | NRP=[last row address - first row address+1]/RSSF | | Number of rows processed |

**Column timing related equations**

| Symbol | Equation | Units | Description |
|---|---|---|---|
| LA | LA=10 | $T_{SCLK}$ | Latency period |
| HB | HB=2 x HBLANK | $T_{SCLK}$ | Horizontal blanking period |
| CSS | {colSS4, colSS}<br>{Reg 27H, bit 0, Reg 1BH, bit 4} | | Column subsampling control |
| CSSF | CSSF=1 if CSS=00<br>CSSF=2 if CSS=X1<br>CSSF=4 if CSS=10 | | Column subsample scale factor |
| NCP | NCP=[last column address-first column address+1]/CSSF | $T_{SCLK}$ | Number of columns processed |
| MINC | Normal mode:<br>MINC>= {ER/CPP- (HB + LA)} x CSSF<br><br>Shutter mode:<br>MINC>=4 | Columns | The minimum allowable number of columns in the image window is a function of the major operating mode and the selected register settings. |

**NOTE**

The number of columns in every window is forced to be a multiple of 4 by the register interface. If not an integer, or a multiple of four, MINC has to be rounded to the next highest multiple of four that satisfies the expression for MINC.

# Exposure Control

Exposure refers to the duration of time between the point at which a pixel is reset and subsequently sampled. Another name is called integration time. This section describes the mechanics of programming the sensor to expose each pixel for a pre-determined period of time. While methods of determining the correct exposure is not within the scope of this section, one should note that the PGA gain settings also factor into determining the correct exposure duration.

By default, pixels in the image sensor array are continuously integrating ambient light. During an image capture process, the period of time that a given row of pixels is integrated starts when an exposure reset is asserted and ends when a sample reset is asserted. The time slot that an exposure reset occurs depends on the operating mode, but is of equal duration in all modes.

## Row Exposure Time

In normal mode, each row is exposed for an equal time duration that is programmed using the exposure registers.

Row Exposure = EXP x $T_{MCLK}$ x RP

where

- EXP is the exposure register settings,
- $T_{MCLK}$ is the input clock period and
- RP is the row processing period (see “Row timing related equations" on page 411)

# A Power-Up Defaults

## Overview

The following table shows the power-up defaults values for the ADCM-2700 camera module.

**Power-up default values table**

| Mode | Default Value | Registers |
|---|---|---|
| Image mode | Video preview mode, QVGA | CONTROL<br>SIZE<br>OUTPUT_FORMAT |
| Frame rate | 15 frames/second | FRAME_RATE |
| Auto exposure | Enabled | EXP_ADJ<br>AF_CTRL1 |
| Auto exposure target value | 64 out of 256 | AE_TARGET |
| Minimum exposure | 50 µs | AE_ETIME_MIN |
| Maximum exposure | 200 ms | AE_ETIME_MAX |
| Default exposure | 10 ms | AE_ETIME_DFLT |
| Deliberate over-exposure | Enabled | AF_CTRL2 |
| Deliberate over-exposure factor | 1.0 | AE_DOE_FACTOR |
| Minimum gain | 1.0 | AE_GAIN_MIN |
| Minimum preferred gain | 2.0 | AE_GAIN_MIN_P |
| Maximum gain | 5.0 | AE_GAIN_MAX |
| Default gain | 2.0 | AE_GAIN_DFLT |

**Power-up default values table (continued)**

| Mode | Default Value | Registers |
|---|---|---|
| Auto white balance: | Enabled | ILLUM<br>AF_CTRL1 |
| Red/green ratio minimum | 0.75 | AWB_RED_MIN |
| Red/green ratio maximum | 1.6484375 | AWB_RED_MAX |
| Red/green ratio default | 1.203125 | AWB_RED_DFLT |
| Blue/green ratio minimum | 0.75 | AWB_BLUE_MIN |
| Blue/green ratio maximum | 2.640625 | AWB_BLUE_MAX |
| Blue/green ratio default | 1.890625 | AWB_BLUE_DFLT |
| AWB green 1 gain | 1.0 | APS_COEF_GRN1 |
| AWB green 2 gain | 1.0 | APS_COEF_GRN2 |
| AWB red gain | 1.0 | APS_COEF_RED |
| AWB blue gain | 1.0 | APS_COEF_BLUE |
| Auto black level | Enabled | AF_CTRL2 |
| Output format | 4:2:2 | |
| Tonemap: | Enabled | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Weighting | Bottom | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Gamma table | sRGB | TM_RED<br>TM_GREEN<br>TM_BLUE |
| Flicker reduction | Enabled | ILLUM<br>AF_CTRL2 |
| Flicker frequency | 50 Hz | AF_CTRL2 |
| Output protocol | CCIR 656 | OUTPUT_CTRL<br>OUTPUT_CTRL_V<br>OUTPUT_CTRL_S |
| Output mirroring | Disabled | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Demosaic | Enabled | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |

**Power-up default values table (continued)**

| Mode | Default Value | | Registers |
|---|---|---|---|
| Color correction matrix coefficients: | Daylight (D65) | | |
| 00 | 2.97265625 | 0x02f9 | CC_COEF_00 |
| 01 | – 0.98828125 | 0x0f03 | CC_COEF_01 |
| 02 | – 0.9921875 | 0x0f02 | CC_COEF_02 |
| 10 | – 0.69140625 | 0x0f4f | CC_COEF_10 |
| 11 | 2.359375 | 0x025c | CC_COEF_11 |
| 12 | – 0.671875 | 0x0f54 | CC_COEF_12 |
| 20 | – 0.125 | 0x0fe0 | CC_COEF_20 |
| 21 | – 1.7109375 | 0x0e4a | CC_COEF_21 |
| 22 | 2.83203125 | 0x02d5 | CC_COEF_22 |
| Color correction matrix pre-offsets: | Daylight (D65) | | |
| 0 | – 8 | 0x01f8 | CC_PRE_OS_0 |
| 1 | – 8 | 0x01f8 | CC_PRE_OS_1 |
| 2 | – 8 | 0x01f8 | CC_PRE_OS_2 |
| Color correction matrix post-offsets: | Daylight (D65) | | |
| 0 | 0 | 0x0000 | CC_POST_OS_0 |
| 1 | 0 | 0x0000 | CC_POST_OS_1 |
| 2 | 0 | 0x0000 | CC_POST_OS_2 |
| Color space conversion coefficients - video mode: | RGB to YCbCr | | |
| 00 | 0.296875 | 0x0026 | CSC_00_V |
| 01 | 0.5859375 | 0x004b | CSC_01_V |
| 02 | 0.1171875 | 0x000f | CSC_02_V |
| 10 | – 0.1584375 | 0x01ed | CSC_10_V |
| 11 | – 0.2890625 | 0x01db | CSC_11_V |
| 12 | 0.4375 | 0x0038 | CSC_12_V |
| 20 | 0.6171875 | 0x004f | CSC_20_V |
| 21 | – 0.515625 | 0x01be | CSC_21_V |
| 22 | – 0.1015625 | 0x01f3 | CSC_22_V |
| Color space conversion offsets - video mode: | | | |
| 0 | 0 | 0x0000 | CSC_OS0_V |
| 1 | 128 | 0x0080 | CSC_OS1_V |
| 2 | 128 | 0x0080 | CSC_OS2_V |

**Power-up default values table (continued)**

| Mode | Default Value | | Registers |
|---|---|---|---|
| Color space conversion coefficients - still mode: | RGB to YCbCr | | |
| 00 | 0.296875 | 0x0026 | CSC_00_S |
| 01 | 0.5859375 | 0x004b | CSC_01_S |
| 02 | 0.1171875 | 0x000f | CSC_02_S |
| 10 | – 0.1584375 | 0x01ed | CSC_10_S |
| 11 | – 0.2890625 | 0x01db | CSC_11_S |
| 12 | 0.4375 | 0x0038 | CSC_12_S |
| 20 | 0.6171875 | 0x004f | CSC_20_S |
| 21 | – 0.515625 | 0x01be | CSC_21_S |
| 22 | – 0.1015625 | 0x01f3 | CSC_22_S |
| Color space conversion offsets - still mode: | | | |
| 0 | 0 | 0x0000 | CSC_OS0_S |
| 1 | 128 | 0x0080 | CSC_OS1_S |
| 2 | 128 | 0x0080 | CSC_OS2_S |
| Color space conversion coefficients - working mode: | RGB to YCbCr | | |
| 00 | 0.296875 | 0x0026 | CSC_COEF_00 |
| 01 | 0.5859375 | 0x004b | CSC_COEF_01 |
| 02 | 0.1171875 | 0x000f | CSC_COEF_02 |
| 10 | – 0.1484375 | 0x01ed | CSC_COEF_10 |
| 11 | – 0.2890625 | 0x01db | CSC_COEF_11 |
| 12 | 0.4375 | 0x0038 | CSC_COEF_12 |
| 20 | 0.6171875 | 0x004f | CSC_COEF_20 |
| 21 | – 0.515625 | 0x01be | CSC_COEF_21 |
| 22 | – 0.1015625 | 0x01f3 | CSC_COEF_22 |
| Color space conversion offsets - working mode: | | | |
| 0 | 0 | 0x0000 | CSC_OS_0 |
| 1 | 128 | 0x0080 | CSC_OS_1 |
| 2 | 128 | 0x0080 | CSC_OS_2 |
| Sizer - video mode: | Enabled, 2:1 QVGA output | | PROC_CTRL_V |
| Input height | 480 | | SZR_IN_HGT_V |
| Input width | 640 | | SZR_IN_WID_V |
| Output height | 240 | | SZR_OUT_HGT_V |
| Output width | 320 | | SZR_OUT_WID_V |
| Sizer - still mode: | Enabled, 1:1 VGA output | | PROC_CTRL_S |
| Input height | 480 | | SZR_IN_HGT_S |
| Input width | 640 | | SZR_IN_WID_S |
| Output height | 480 | | SZR_OUT_HGT_S |
| Output width | 640 | | SZR_OUT_WID_S |

**Power-up default values table (continued)**

| Mode | Default Value | | Registers |
|---|---|---|---|
| Sizer - working mode: | Enabled, 2:1 QVGA output | | PROCESS_CTRL |
| Input height | 480 | | SZR_IN_H |
| Input width | 640 | | SZR_IN_W |
| Output height | 240 | | SZR_OUT_H |
| Output width | 320 | | SZR_OUT_W |
| CCIR 656: | | | |
| VSYNC to HSYNC hold time | 0 | | CCIR_TIMING |
| VSYNC to HSYNC setup time | 0 | | CCIR_TIMING |
| Embedded VSYNC to HSYNC | Disabled | | OUTPUT_CTRL<br>OUTPUT_CTRL_V<br>OUTPUT_CTRL_S |
| Number of dummy HSYNC | 0 | | CCIR_TIMING |
| Luminance (or red) clipping: | | | |
| Maximum, minimum | 255, 0 | | R_Y_MAX_MIN |
| Chrominance, Cb (or green) clipping: | | | |
| Maximum, minimum | 255, 0 | | G_CB_MAX_MIN |
| Chrominance, Cr (or blue) clipping: | | | |
| Maximum, minimum | 255, 0 | | B_CR_MAX_MIN |
| Test Generator: | Disabled | | DATA_GEN |
| Mode | Normal mode | | DATA_GEN |
| Image size register (read only): | | | |
| Width | 0 | | I_WIDTH |
| Height | 0 | | I_HEIGHT |
| Start of frame and end of frame codes | Disabled | | OUTPUT_CTRL<br>OUTPUT_CTRL_V<br>OUTPUT_CTRL_S |
| Start of frame code | Video = 0xff | Still = 0xfe | SOF_CODES<br>SOF_CODE_W |
| End of frame codes | Good = 0x00 | Bad = 0x01 | EOF_CODES<br>EOF_CODES_W |
| PLL: | Disabled | | PLL_CTRL |
| Large divisors | REF = 29 | VCO = 9 | PLL_DIV_L |
| Small divisors | REF = 2 | VCO = 0 | PLL_DIV_S |
| Clocks: | | | |
| Input clock frequency | 13,000 | KHz | CLK_PER |
| Sensor clocks per pixel | Video = 3 | Still = 3 | CLK_PIXEL<br>CPP_V<br>CPP_S |

**Power-up default values table (continued)**

| Mode | Default Value | | Registers |
|---|---|---|---|
| Clock divisors: | | | |
| Control | 1 | | CTL_CLK_DIV |
| Image sensor | 1 | | SEN_CLK_DIV |
| Image pipeline | 1 | | IP_CLK_DIV |
| Output VCLK | Video = 6 | Still = 1 | PARALLEL_CTRL<br>PARALLEL_CTRL_V<br>PARALLEL_CTRL_S |
| Anti-vignetting: | Disabled | | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Top left | FWROW = 1 | FWCOL = 1 | AV_LEFT_TOP |
| Bottom right | LWROW = 488 | LWCOL = 648 | AV_RIGHT_BOT |
| Center row | 248 | | AV_CENTER_ROW |
| Center column | 348 | | AV_CENTER_COL |
| Oval factor | 1.0 | | AV_OVAL_FACT |
| Noise adaptive color correction | Disabled | | AF_CTRL1 |
| Statistics: | Linear, disabled | | STAT_CAP_CTRL |
| Mode control | Full weighting | | STAT_MODE_CTRL |
| Red pixel sum | 0 | | RED_SUM |
| Green 1 pixel sum | 0 | | GREEN1_SUM |
| Green 2 pixel sum | 0 | | GREEN2_SUM |
| Blue pixel sum | 0 | | BLUE_SUM |
| Flicker statistics | Disabled | | STAT_CAP_CTRL |
| Halftoning | Bypassed | | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Sharpening | No effect | | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |
| Bad pixel correction | Enabled | | PROCESS_CTRL<br>PROC_CTRL_V<br>PROC_CTRL_S |

# B
# Register List



The following tables list the registers, first numerically by address and then alphabetically by mnemonic.

## By Address

**Registers by Address**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0000 | ID | Chip ID | 0x0060 | 139 |
| 0x0002 | CONTROL | Camera control | 0x0001 | 140 |
| 0x0004 | STATUS | Camera status | 0x0004 | 142 |
| 0x0006 | CLK_FREQ | Input clock frequency | 0x32c8 | 144 |
| 0x0008 | SIZE | Image size and orientation | 0x0605 | 145 |
| 0x000a | OUTPUT_FORMAT | Output format | 0x0909 | 147 |
| 0x000c | EXPOSURE | Exposure | 0x03e8 | 149 |
| 0x000e | EXP_ADJ | Exposure adjustment | 0x0000 | 150 |
| 0x0010 | ILLUM | Illumination | 0x0000 | 151 |
| 0x0012 | FRAME_RATE | Requested frame rate | 0x0096 | 152 |
| 0x0014 | Reserved | Reserved | | |
| 0x0016 | A_FRAME_RATE | Actual frame rate | 0x0096 | 154 |
| 0x0018 | SENSOR_WID_V | Sensor window width, video mode | 0x0000 | 155 |
| 0x001a | SENSOR_HGT_V | Sensor window height, video mode | 0x0000 | 156 |
| 0x001c | OUTPUT_WID_V | Output window width, video mode | 0x0000 | 157 |
| 0x001e | OUTPUT_HGT_V | Output window height, video mode | 0x0000 | 158 |
| 0x0020 | SENSOR_WID_S | Sensor window width, still mode | 0x0000 | 159 |
| 0x0022 | SENSOR_HGT_S | Sensor window height, still mode | 0x0000 | 160 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0024 | OUTPUT_WID_V | Output window width, still mode | 0x0000 | 161 |
| 0x0026 | OUTPUT_HGT_S | Output window height, still mode | 0x0000 | 162 |
| 0x0028 to 0x007e | Reserved | Reserved | | |
| 0x0080 | I_CLK_DIV | Initial clock divider | 0x0001 | 166 |
| 0x0082 | CTL_CLK_DIV | Clock dividers for control and serial interfaces | 0x4000 | 167 |
| 0x0084 | SEN_CLK_DIV | Sensor clock dividers | 0x0000 | 168 |
| 0x0086 | IP_CLK_DIV | Clock dividers for image pipeline | 0x0000 | 169 |
| 0x0088 | TST_MODE | Latched test mode | 0x0000 | 170 |
| 0x008a | SER_ADDR | Serial interface device address | 0x0053 | 171 |
| 0x008c | SER_PARM | Serial Interface parameters | 0x0000 | 172 |
| 0x008e | OUT_CTRL | Output control | 0x0000 | 173 |
| 0x0090 | PLL_CTRL | PLL control | 0x0024 | 174 |
| 0x0092 | PLL_DIV_L | PLL divisors, large values | 0x1d09 | 175 |
| 0x0094 | PLL_DIV_S | PLL divisors, small values | 0x0200 | 176 |
| 0x0096 to 0x00fe | Reserved | Reserved | | |
| 0x0100 | SZR_IN_WID_V | Sizer input width, video mode | 0x0280 | 184 |
| 0x0102 | SZR_IN_HGT_V | Sizer input height, video mode | 0x01e0 | 185 |
| 0x0104 | SZR_OUT_WID_V | Sizer output width, video mode | 0x0140 | 186 |
| 0x0106 | SZR_OUT_HGT_V | Sizer output height, video mode | 0x00f0 | 187 |
| 0x0108 | CPP_V | Clocks per pixel, video mode | 0x0002 | 188 |
| 0x010a | HBLANK_V | Horizontal blanking period, video mode | 0x0000 | 189 |
| 0x010c | VBLANK_V | Vertical blanking period, video mode | 0x0000 | 190 |
| 0x010e | MIN_MAX_F_V | Frame convergence rates, video mode | 0x0000 | 191 |
| 0x0110 | OUTPUT_CTRL_V | Output control, video mode | 0x9019 | 192 |
| 0x0112 | PROC_CTRL_V | Processing control, video mode | 0x0280 | 194 |
| 0x0114 | RPT_V | Row processing time, video mode | 0x0546 | 196 |
| 0x0116 | HYSNC_PER_V | HSYNC period, video mode | 0x0a8b | 197 |
| 0x0118 | CLK_DIV_V | Clock divisors, video mode | 0x0000 | 198 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x011a | PARALLEL_CTRL_V | Parallel output control, video mode | 0x0003 | 199 |
| 0x011c | SEN_CTRL_V | Sensor control, video mode | 0x0000 | 200 |
| 0x011e | Reserved | Reserved | | |
| 0x0120 | SZR_IN_WID_S | Sizer input width, still mode | 0x0280 | 202 |
| 0x0122 | SZR_IN_HGT_S | Sizer input height, still mode | 0x01e0 | 203 |
| 0x0124 | SZR_OUT_WID_S | Sizer output width, still mode | 0x0280 | 204 |
| 0x0126 | SZR_OUT_HGT_S | Sizer output height, still mode | 0x01e0 | 205 |
| 0x0128 | CPP_S | Clocks per pixel, still mode | 0x0002 | 206 |
| 0x012a | HBLANK_S | Horizontal blanking period, still mode | 0x0000 | 207 |
| 0x012c | VBLANK_S | Vertical blanking period, still mode | 0x0000 | 208 |
| 0x012e | MIN_MAX_F_S | Frame convergence rates, still mode | 0x0002 | 209 |
| 0x0130 | OUTPUT_CTRL_S | Output control, still mode | 0x8019 | 210 |
| 0x0132 | PROC_CTRL_S | Processing control, still mode | 0x0280 | 212 |
| 0x0134 | RPT_S | Row processing time, still mode | 0x0546 | 214 |
| 0x0136 | HYSNC_PER_S | HSYNC period, still mode | 0x0545 | 215 |
| 0x0138 | CLK_DIV_S | Clock divisors, still mode | 0x0000 | 216 |
| 0x013a | PARALLEL_CTRL_S | Parallel output control, still mode | 0x0000 | 217 |
| 0x013c | SEN_CTRL_S | Sensor control, still mode | 0x0000 | 218 |
| 0x013e | Reserved | Reserved | | |
| 0x0140 | AF_CTRL1 | Auto functions control 1 | 0x0013 | 220 |
| 0x0142 | AF_CTRL2 | Auto functions control 2 | 0x0001 | 221 |
| 0x0144 | AF_STATUS | Auto functions status | 0x0000 | 222 |
| 0x0146 | SOF_CODES | Start of frame codes | 0xfeff | 223 |
| 0x0148 | EOF_CODES | End of frame codes | 0x0100 | 224 |
| 0x014a | ABL_TARGET | Auto black level target | 0x0005 | 225 |
| 0x014c | ABL_MAX_BIN | Auto black level maximum bin | 0x0003 | 226 |
| 0x014e | ABL_MAX_BLACK | auto black level maximum black | 0x0010 | 227 |
| 0x0150 | AE_GAIN_MIN | Auto exposure gain minimum | 0x01c0 | 228 |
| 0x0152 | AE_GAIN_MIN_P | Auto exposure gain minimum, preferred | 0x0200 | 229 |
| 0x0154 | AE_GAIN_MAX | Auto exposure gain maximum | 0x0500 | 230 |
| 0x0156 | AE_GAIN_DFLT | Auto exposure gain default | 0x0200 | 231 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0158 | AE_ETIME_MIN | Auto exposure time minimum | 0x0005 | 232 |
| 0x015a | AE_ETIME_MAX | Auto exposure time maximum | 0x4e20 | 233 |
| 0x015c | AE_ETIME_DFLT | Auto exposure time default | 0x03e8 | 234 |
| 0x015e | AE_TARGET | Auto exposure target | 0x0040 | 235 |
| 0x0160 | AE_TOL_ACQ | Auto exposure tolerance acquire | 0x0118 | 236 |
| 0x0162 | AE_TOL_MON | Auto exposure tolerance monitor | 0x0118 | 237 |
| 0x0164 | AE_MARGIN | Auto exposure margin | 0x0120 | 238 |
| 0x0166 | AE_DOE_FACTOR | AE deliberate overexposure factor | 0x014e | 239 |
| 0x0168 | AE_DOE_MARGIN | AE deliberate overexposure margin | 0x0140 | 240 |
| 0x016a to 0x016e | Reserved | Reserved | | |
| 0x0170 | AWB_RED_MIN | AWB minimum red/green ratio | 0x00c0 | 242 |
| 0x0172 | AWB_RED_MAX | AWB maximum red/green ratio | 0x01a6 | 243 |
| 0x0174 | AWB_RED_DFLT | AWB default red/green ratio | 0x0134 | 244 |
| 0x0176 | AWB_BLUE_MIN | AWB minimum blue/green ratio | 0x00c0 | 245 |
| 0x0178 | AWB_BLUE_MAX | AWB maximum blue/green ratio | 0x02a4 | 246 |
| 0x017a | AWB_BLUE_DFLT | AWB default blue/green ratio | 0x01e4 | 247 |
| 0x017c | AWB_TOL_ACQ | Auto white balance tolerance acquire | 0x0110 | 248 |
| 0x017e | AWB_TOL_MON | Auto white balance tolerance monitor | 0x0120 | 249 |
| 0x0180 | FIRMWARE_REV | Current firmware revision | 0x0152 | 250 |
| 0x0182 | FLICK_CFG_1 | Flicker configuration 1 | 0x2aeb | 251 |
| 0x0184 | FLICK_CFG_2 | Flicker configuration 2 | 0x0005 | 252 |
| 0x0186 0x0188 | Reserved | Reserved | | |
| 0x018a | MAX_SCLK | Maximum sensor clock | 0x1964 | 254 |
| 0x018c 0x018e | Reserved | Reserved | | |
| 0x0190 | CSC_00_V | Color space conversion coefficient 00, video | 0x0026 | 256 |
| 0x0192 | CSC_01_V | Color space conversion coefficient 01, video | 0x004b | 256 |
| 0x0194 | CSC_02_V | Color space conversion coefficient 02, video | 0x000f | 256 |
| 0x0196 | CSC_10_V | Color space conversion coefficient 10, video | 0x01ed | 256 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0198 | CSC_11_V | Color space conversion coefficient 11, video | 0x01db | 256 |
| 0x019a | CSC_12_V | Color space conversion coefficient 12, video | 0x0038 | 256 |
| 0x019c | CSC_20_V | Color space conversion coefficient 20, video | 0x004f | 256 |
| 0x019e | CSC_21_V | Color space conversion coefficient 21, video | 0x01be | 256 |
| 0x01a0 | CSC_22_V | Color space conversion coefficient 22, video | 0x01f3 | 256 |
| 0x01a2 | CSC_OS0_V | Color space conversion offset 0, video | 0x0000 | 257 |
| 0x01a4 | CSC_OS1_V | Color space conversion offset 1, video | 0x0080 | 257 |
| 0x01a6 | CSC_OS2_V | Color space conversion offset 2, video | 0x0080 | 257 |
| 0x01a8 | CSC_00_S | Color space conversion coefficient 00, still | 0x0026 | 258 |
| 0x01aa | CSC_01_S | Color space conversion coefficient 01, still | 0x004b | 258 |
| 0x01ac | CSC_02_S | Color space conversion coefficient 02, still | 0x000f | 258 |
| 0x01ae | CSC_10_S | Color space conversion coefficient 10, still | 0x01ed | 258 |
| 0x01b0 | CSC_11_S | Color space conversion coefficient 11, still | 0x01db | 258 |
| 0x01b2 | CSC_12_S | Color space conversion coefficient 12, still | 0x0038 | 258 |
| 0x01b4 | CSC_20_S | Color space conversion coefficient 20, still | 0x004f | 258 |
| 0x01b6 | CSC_21_S | Color space conversion coefficient 21, still | 0x01be | 258 |
| 0x01b8 | CSC_22_S | Color space conversion coefficient 22, still | 0x01f3 | 258 |
| 0x01ba | CSC_OS0_S | Color space conversion offset 0, still | 0x0000 | 259 |
| 0x01bc | CSC_OS1_S | Color space conversion offset 1, still | 0x0080 | 259 |
| 0x01be | CSC_OS2_S | Color space conversion offset 2, still | 0x0080 | 259 |
| 0x01c0 | TM_COEF_00_V | Tonemap coefficient 00, video | 0x0000 | 261 |
| 0x01c2 | TM_COEF_01_V | Tonemap coefficient 01, video | 0x0017 | 261 |
| 0x01c4 | TM_COEF_02_V | Tonemap coefficient 02, video | 0x0032 | 261 |
| 0x01c6 | TM_COEF_03_V | Tonemap coefficient 03, video | 0x0046 | 261 |
| 0x01c8 | TM_COEF_04_V | Tonemap coefficient 04, video | 0x0056 | 261 |
| 0x01ca | TM_COEF_05_V | Tonemap coefficient 05, video | 0x0064 | 261 |
| 0x01cc | TM_COEF_06_V | Tonemap coefficient 06, video | 0x0071 | 261 |
| 0x01ce | TM_COEF_07_V | Tonemap coefficient 07, video | 0x007c | 261 |
| 0x01d0 | TM_COEF_08_V | Tonemap coefficient 08, video | 0x0086 | 261 |
| 0x01d2 | TM_COEF_09_V | Tonemap coefficient 09, video | 0x0099 | 261 |
| 0x01d4 | TM_COEF_10_V | Tonemap coefficient 10, video | 0x00a9 | 261 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x01d6 | TM_COEF_11_V | Tonemap coefficient 11, video | 0x00b8 | 261 |
| 0x01d8 | TM_COEF_12_V | Tonemap coefficient 12, video | 0x00c6 | 261 |
| 0x01da | TM_COEF_13_V | Tonemap coefficient 13, video | 0x00df | 261 |
| 0x01dc | TM_COEF_14_V | Tonemap coefficient 14, video | 0x00f5 | 261 |
| 0x01de | TM_COEF_15_V | Tonemap coefficient 15, video | 0x0109 | 261 |
| 0x01e0 | TM_COEF_16_V | Tonemap coefficient 16, video | 0x011b | 261 |
| 0x01e2 | TM_COEF_17_V | Tonemap coefficient 17, video | 0x013d | 261 |
| 0x01e4 | TM_COEF_18_V | Tonemap coefficient 18, video | 0x015a | 261 |
| 0x01e6 | TM_COEF_19_V | Tonemap coefficient 19, video | 0x0175 | 261 |
| 0x01e8 | TM_COEF_20_V | Tonemap coefficient 20, video | 0x018d | 261 |
| 0x01ea | TM_COEF_21_V | Tonemap coefficient 21, video | 0x01ba | 261 |
| 0x01ec | TM_COEF_22_V | Tonemap coefficient 22, video | 0x01e1 | 261 |
| 0x01ee | TM_COEF_23_V | Tonemap coefficient 23, video | 0x0205 | 261 |
| 0x01f0 | TM_COEF_24_V | Tonemap coefficient 24, video | 0x0225 | 261 |
| 0x01f2 | TM_COEF_25_V | Tonemap coefficient 25, video | 0x0261 | 261 |
| 0x01f4 | TM_COEF_26_V | Tonemap coefficient 26, video | 0x0295 | 261 |
| 0x01f6 | TM_COEF_27_V | Tonemap coefficient 27, video | 0x02c5 | 261 |
| 0x01f8 | TM_COEF_28_V | Tonemap coefficient 28, video | 0x02f1 | 261 |
| 0x01fa | TM_COEF_29_V | Tonemap coefficient 29, video | 0x033f | 261 |
| 0x01fc | TM_COEF_30_V | Tonemap coefficient 30, video | 0x0385 | 261 |
| 0x01fe | TM_COEF_31_V | Tonemap coefficient 31, video | 0x03c5 | 261 |
| 0x0200 | TM_COEF_32_V | Tonemap coefficient 32, video | 0x0400 | 261 |
| 0x0202 | TM_COEF_00_S | Tonemap coefficient 00, still | 0x0000 | 262 |
| 0x0204 | TM_COEF_01_S | Tonemap coefficient 01, still | 0x0017 | 262 |
| 0x0206 | TM_COEF_02_S | Tonemap coefficient 02, still | 0x0032 | 262 |
| 0x0208 | TM_COEF_03_S | Tonemap coefficient 03, still | 0x0046 | 262 |
| 0x020a | TM_COEF_04_S | Tonemap coefficient 04, still | 0x0056 | 262 |
| 0x020c | TM_COEF_05_S | Tonemap coefficient 05, still | 0x0064 | 262 |
| 0x020e | TM_COEF_06_S | Tonemap coefficient 06, still | 0x0071 | 262 |
| 0x0210 | TM_COEF_07_S | Tonemap coefficient 07, still | 0x007c | 262 |
| 0x0212 | TM_COEF_08_S | Tonemap coefficient 08, still | 0x0086 | 262 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0214 | TM_COEF_09_S | Tonemap coefficient 09, still | 0x0099 | 262 |
| 0x0216 | TM_COEF_10_S | Tonemap coefficient 10, still | 0x00a9 | 262 |
| 0x0218 | TM_COEF_11_S | Tonemap coefficient 11, still | 0x00b8 | 262 |
| 0x021a | TM_COEF_12_S | Tonemap coefficient 12, still | 0x00c6 | 262 |
| 0x021c | TM_COEF_13_S | Tonemap coefficient 13, still | 0x00df | 262 |
| 0x021e | TM_COEF_14_S | Tonemap coefficient 14, still | 0x00f5 | 262 |
| 0x0220 | TM_COEF_15_S | Tonemap coefficient 15, still | 0x0109 | 262 |
| 0x0222 | TM_COEF_16_S | Tonemap coefficient 16, still | 0x011b | 262 |
| 0x0224 | TM_COEF_17_S | Tonemap coefficient 17, still | 0x013d | 262 |
| 0x0226 | TM_COEF_18_S | Tonemap coefficient 18, still | 0x015a | 262 |
| 0x0228 | TM_COEF_19_S | Tonemap coefficient 19, still | 0x0175 | 262 |
| 0x022a | TM_COEF_20_S | Tonemap coefficient 20, still | 0x018d | 262 |
| 0x022c | TM_COEF_21_S | Tonemap coefficient 21, still | 0x01ba | 262 |
| 0x022e | TM_COEF_22_S | Tonemap coefficient 22, still | 0x01e1 | 262 |
| 0x0230 | TM_COEF_23_S | Tonemap coefficient 23, still | 0x0205 | 262 |
| 0x0232 | TM_COEF_24_S | Tonemap coefficient 24, still | 0x0225 | 262 |
| 0x0234 | TM_COEF_25_S | Tonemap coefficient 25, still | 0x0261 | 262 |
| 0x0236 | TM_COEF_26_S | Tonemap coefficient 26, still | 0x0295 | 262 |
| 0x0238 | TM_COEF_27_S | Tonemap coefficient 27, still | 0x02c5 | 262 |
| 0x023a | TM_COEF_28_S | Tonemap coefficient 28, still | 0x02f1 | 262 |
| 0x023c | TM_COEF_29_S | Tonemap coefficient 29, still | 0x033f | 262 |
| 0x023e | TM_COEF_30_S | Tonemap coefficient 30, still | 0x0385 | 262 |
| 0x0240 | TM_COEF_31_S | Tonemap coefficient 31, still | 0x03c5 | 262 |
| 0x0242 | TM_COEF_32_S | Tonemap coefficient 32, still | 0x0400 | 262 |
| 0x0244 to 0x024e | Reserved | Reserved | | |
| 0x0250 | NACC_EGP_1 | Noise adaptive color correction, EGP 1 | 0x05dc | 265 |
| 0x0252 | NACC_SAT_1 | Noise adaptive color correction, saturation 1 | 0x0000 | 265 |
| 0x0254 | NACC_EGP_2 | Noise adaptive color correction, EGP 2 | 0x0465 | 265 |
| 0x0256 | NACC_SAT_2 | Noise adaptive color correction, saturation 2 | 0x0040 | 265 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0258 | NACC_EGP_3 | Noise adaptive color correction, EGP 3 | 0x02ee | 265 |
| 0x025a | NACC_SAT_3 | Noise adaptive color correction, saturation 3 | 0x0080 | 265 |
| 0x025c | NACC_EGP_4 | Noise adaptive color correction, EGP 4 | 0x0177 | 265 |
| 0x025e | NACC_SAT_4 | Noise adaptive color correction, saturation 4 | 0x00c0 | 265 |
| 0x0260 | NACC_EGP_5 | Noise adaptive color correction, EGP 5 | 0x0000 | 265 |
| 0x0262 | NACC_SAT_5 | Noise adaptive color correction, saturation 5 | 0x0100 | 265 |
| 0x0264 | NACC_EGP_6 | Noise adaptive color correction, EGP 6 | 0x0000 | 265 |
| 0x0266 | NACC_SAT_6 | Noise adaptive color correction, saturation 6 | 0x0000 | 265 |
| 0x0268 | NACC_EGP_7 | Noise adaptive color correction, EGP 7 | 0x0000 | 265 |
| 0x026a | NACC_SAT_7 | Noise adaptive color correction, saturation 7 | 0x0000 | 265 |
| 0x026c | NACC_EGP_8 | Noise adaptive color correction, EGP 8 | 0x0000 | 265 |
| 0x026e | NACC_SAT_8 | Noise adaptive color correction, saturation 8 | 0x0000 | 265 |
| 0x0270 | NACC_BC_00 | NACC bright coefficients 00 | 0x0235 | 266 |
| 0x0272 | NACC_BC_01 | NACC bright coefficients 01 | 0xff46 | 266 |
| 0x0274 | NACC_BC_02 | NACC bright coefficients 02 | 0xff85 | 266 |
| 0x0276 | NACC_BC_10 | NACC bright coefficients 10 | 0xff64 | 266 |
| 0x0278 | NACC_BC_11 | NACC bright coefficients 11 | 0x01fc | 266 |
| 0x027a | NACC_BC_12 | NACC bright coefficients 12 | 0xff9f | 266 |
| 0x027c | NACC_BC_20 | NACC bright coefficients 20 | 0x0008 | 266 |
| 0x027e | NACC_BC_21 | NACC bright coefficients 21 | 0xfe8d | 266 |
| 0x0280 | NACC_BC_22 | NACC bright coefficients 22 | 0x026b | 266 |
| 0x0282 | NACC_DC_00 | NACC dark coefficients 00 | 0x0048 | 266 |
| 0x0284 | NACC_DC_01 | NACC dark coefficients 01 | 0x010b | 266 |
| 0x0286 | NACC_DC_02 | NACC dark coefficients 02 | 0xffaa | 266 |
| 0x0288 | NACC_DC_10 | NACC dark coefficients 10 | 0x0048 | 266 |
| 0x028a | NACC_DC_11 | NACC dark coefficients 11 | 0x010b | 266 |
| 0x028c | NACC_DC_12 | NACC dark coefficients 12 | 0xffaa | 266 |
| 0x028e | NACC_DC_20 | NACC dark coefficients 20 | 0x0048 | 266 |
| 0x0290 | NACC_DC_21 | NACC dark coefficients 21 | 0x010b | 266 |
| 0x0292 | NACC_DC_22 | NACC dark coefficients 22 | 0xffaa | 266 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x0294 to 0x07fe | Reserved | Reserved | | |
| 0x0800 | IDENT | Image sensor identification | 0x60 | 272 |
| 0x0801 | IS_STATUS | Image sensor status | 0x00 | 273 |
| 0x0802 0x0804 | Reserved | Reserved | | |
| 0x0805 | ICTRL | Interface control | 0x00 | 275 |
| 0x0806 0x0807 0x0808 | Reserved | Reserved | | |
| 0x0809 | ADC_CTRL | ADC control | 0x01 | 277 |
| 0x080a | FWROW | Window first row address | 0x01 | 278 |
| 0x080b | FWCOL | Window first column address | 0x01 | 279 |
| 0x080c | LWROW | Window last row address | 0x7a | 280 |
| 0x080d | LWCOL | Window last column address | 0xa2 | 281 |
| 0x080e | CLK_PIXEL | Clocks per pixel | 0x02 | 282 |
| 0x080f | EREC_PGA | Even row, even column (green 1) PGA gain | 0x00 | 283 |
| 0x0810 | EROC_PGA | Even row, odd column (red) PGA gain | 0x00 | 284 |
| 0x0811 | OREC_PGA | Odd row, even column (blue) PGA gain | 0x00 | 285 |
| 0x0812 | OROC_PGA | Odd row, odd column (green 2) PGA gain | 0x00 | 286 |
| 0x0813 | ROWEXP_L | Row exposure low | 0x54 | 287 |
| 0x0814 | ROWEXP_H | Row exposure high | 0x00 | 288 |
| 0x0815 | SROWEXP | Sub row exposure | 0x31 | 289 |
| 0x0816 | ERROR | Error control | 0x00 | 290 |
| 0x0817 0x0818 | Reserved | Reserved | | |
| 0x0819 | HBLANK | Horizontal blank | 0x00 | 292 |
| 0x081a | VBLANK | Vertical blank | 0x00 | 293 |
| 0x081b | CONFIG_1 | Image sensor configuration 1 | 0x0e | 294 |
| 0x081c | CONTROL_1 | Image sensor control 1 | 0x24 | 295 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x081d to 0x0826 | Reserved | Reserved | | |
| 0x0827 | CONFIG_2 | Image sensor configuration 2 | 0x00 | 298 |
| 0x0828 | GRR_CTRL | Ground reset reference control | 0x00 | 299 |
| 0x0829 to 0x0836 | Reserved | Reserved | | |
| 0x0837 | BIAS_TRM | Bias trim | 0x00 | 301 |
| 0x0838 to 0x08d6 | Reserved | Reserved | 0x00 | |
| 0x08d7 | SMP_GR_E2 | Sample ground reference edge 2 | 0x00 | 303 |
| 0x08d8 | SMP_GR_E1 | Sample ground reference edge 1 | 0x10 | 304 |
| 0x08d9 | SMP_GR_E0 | Sample ground reference edge 0 | 0x0a | 305 |
| 0x08da 0x08db | Reserved | Reserved | | |
| 0x08dc | EXP_GR_E1 | Exposure, ground reference edge 1 | 0x10 | 307 |
| 0x08dd | EXP_GR_E0 | Exposure, ground reference edge 0 | 0x06 | 308 |
| 0x08de | Reserved | Reserved | | |
| 0x08df | GR_POL | Ground reference polarity | 0xd3 | 310 |
| 0x08e0 to 0x08ea | Reserved | Reserved | | |
| 0x08eb | SMP_RST_E2 | Sample, reset edge 2 | 0x04 | 312 |
| 0x08ec | SMP_RST_E1 | Sample, reset edge 1 | 0x10 | 313 |
| 0x08ed | SMP_RST_E0 | Sample, reset edge 0 | 0x07 | 314 |
| 0x08ee 0x08ef | Reserved | Reserved | | |
| 0x08f0 | EXP_RST_E1 | Exposure, reset edge 1 | 0x10 | 316 |
| 0x08f1 | EXP_RST_E0 | Exposure, reset edge 1 | 0x03 | 317 |
| 0x08f2 | Reserved | Reserved | 0x00 | |
| 0x08f3 | RESET_POL | Reset polarity enable | 0xd3 | 319 |
| 0x08f4 | Reserved | Reserved | | |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x08f5 | SMP_PRST_E2 | Sample, preset edge 2 | 0x00 | 321 |
| 0x08f6 | SMP_PRST_E1 | Sample, preset edge 1 | 0x02 | 322 |
| 0x08f7 | SMP_PRST_E0 | Sample, preset edge 0 | 0x0a | 323 |
| 0x08f8<br>0x08f9 | Reserved | Reserved | | |
| 0x08fa | EXP_PRST_E1 | Exposure, preset edge 1 | 0x02 | 325 |
| 0x08fb | EXP_PRST_E0 | Exposure, preset edge 1 | 0x06 | 326 |
| 0x08fc | Reserved | Reserved | | |
| 0x08fd | PRESET_POL | Preset polarity enable | 0xd3 | 328 |
| 0x08fe<br>to<br>0x1000 | Reserved | Reserved | | |
| 0x1002 | CMD_1 | Main command 1 | 0x0000 | 335 |
| 0x1004 | CMD_2 | Main command 2 (write 1's only) | 0x0002 | 336 |
| 0x1006 | Reserved | Reserved | | |
| 0x1008 | OUTPUT_CTRL | Output control, working | 0x9019 | 338 |
| 0x100a | PARALLEL_CTRL | Parallel output control working copy | 0x0000 | 340 |
| 0x100c | SOF_CODE_W | Start of frame code working copy | 0x00ff | 341 |
| 0x100e | EOF_CODES | End of frame codes working copy | 0x0100 | 342 |
| 0x1010 | CCIR_TIMING | CCIR interface timing | 0x0000 | 343 |
| 0x1012 | R_Y_MAX_MIN | Luminance, Y (or red) maximum/minimum | 0xff00 | 344 |
| 0x1014 | G_CB_MAX_MIN | Chrominance, Cb (or green) maximum/minimum | 0xff00 | 345 |
| 0x1016 | B_CR_MAX_MIN | Chrominance, Cr (or blue) maximum/minimum | 0xff00 | 346 |
| 0x1018 | PROCESS_CTRL | Processing control working copy | 0x0280 | 347 |
| 0x101a | BPA_SF_GTHRESH | BPA scale factor, green filter threshold | 0x0220 | 349 |
| 0x101c | BPA_OUTL_PED | BPA outlier, pedestal | 0x4008 | 350 |
| 0x101e | BPA_BADPIX_CNT | BPA bad pixel count (read only) | 0x0000 | 351 |
| 0x1020 | SZR_IN_W | Sizer input width | 0x0280 | 352 |
| 0x1022 | SZR_IN_H | Sizer input height | 0x01e0 | 353 |
| 0x1024 | SZR_OUT_W | Sizer output width | 0x0140 | 354 |
| 0x1026 | SZR_OUT_H | Sizer output height | 0x00f0 | 355 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x1028 | CC_COEF_00 | Color correction coefficient 00 | 0x02f9 | 358 |
| 0x102a | CC_COEF_01 | Color correction coefficient 01 | 0x0f03 | 358 |
| 0x102c | CC_COEF_02 | Color correction coefficient 02 | 0x0f02 | 358 |
| 0x102e | CC_COEF_10 | Color correction coefficient 10 | 0x0f4f | 358 |
| 0x1030 | CC_COEF_11 | Color correction coefficient 11 | 0x025c | 358 |
| 0x1032 | CC_COEF_12 | Color correction coefficient 12 | 0x0f54 | 358 |
| 0x1034 | CC_COEF_20 | Color correction coefficient 20 | 0x0fe0 | 358 |
| 0x1036 | CC_COEF_21 | Color correction coefficient 21 | 0x0e4a | 358 |
| 0x1038 | CC_COEF_22 | Color correction coefficient 22 | 0x02d5 | 358 |
| 0x103a | CC_PRE_OS_0 | Color correction pre-offset 0 | 0x01f8 | 360 |
| 0x103c | CC_PRE_OS_1 | Color correction pre-offset 1 | 0x01f8 | 360 |
| 0x103e | CC_PRE_OS_2 | Color correction pre-offset 2 | 0x01f8 | 360 |
| 0x1040 | CC_POST_OS_0 | Color correction post-offset 0 | 0x0000 | 360 |
| 0x1042 | CC_POST_OS_1 | Color correction post-offset 1 | 0x0000 | 360 |
| 0x1044 | CC_POST_OS_2 | Color correction post-offset 2 | 0x0000 | 360 |
| 0x1046 | CSC_COEF_00 | Color space conversion coefficient 00 | 0x0026 | 363 |
| 0x1048 | CSC_COEF_01 | Color space conversion coefficient 01 | 0x004b | 363 |
| 0x104a | CSC_COEF_02 | Color space conversion coefficient 02 | 0x000f | 363 |
| 0x104c | CSC_COEF_10 | Color space conversion coefficient 10 | 0x01ed | 363 |
| 0x104e | CSC_COEF_11 | Color space conversion coefficient 11 | 0x01db | 363 |
| 0x1050 | CSC_COEF_12 | Color space conversion coefficient 12 | 0x0038 | 363 |
| 0x1052 | CSC_COEF_20 | Color space conversion coefficient 20 | 0x004f | 363 |
| 0x1054 | CSC_COEF_21 | Color space conversion coefficient 21 | 0x01be | 363 |
| 0x1056 | CSC_COEF_22 | Color space conversion coefficient 22 | 0x01f3 | 363 |
| 0x1058 | CSC_OS_0 | Color space conversion offset 0 | 0x0000 | 364 |
| 0x105a | CSC_OS_1 | Color space conversion offset 1 | 0x0080 | 364 |
| 0x105c | CSC_OS_2 | Color space conversion offset 2 | 0x0080 | 364 |
| 0x105e | DATA_GEN | Test data generator | 0x0000 | 365 |
| 0x1060 | HSYNC_PER | Horizontal synchronization period | 0x0a8b | 366 |
| 0x1062 | APS_COEF_GRN1 | Green 1 AWB gain | 0x0080 | 368 |
| 0x1064 | APS_COEF_RED | Red AWB gain | 0x0080 | 368 |

**Registers by Address (continued)**

| Address | Mnemonic | Register name | Default | Page |
|---|---|---|---|---|
| 0x1066 | APS_COEF_BLUE | Blue AWB gain | 0x0080 | 368 |
| 0x1068 | APS_COEF_GRN2 | Green 2 AWB gain | 0x0080 | 368 |
| 0x106a | AV_LEFT_TOP | Anti-vignetting, sensor first row and column | 0x0101 | 369 |
| 0x106c | AV_RIGHT_BOT | Anti-vignetting, sensor last row and column | 0xa27a | 370 |
| 0x106e | AV_CENTER_COL | Anti-vignetting, sensor center column | 0x0148 | 371 |
| 0x1070 | AV_CENTER_ROW | Anti-vignetting, sensor center row | 0x00f8 | 372 |
| 0x1072 | STAT_CAP_CTRL | Image statistics capture control | 0x0021 | 373 |
| 0x1074 | STAT_MODE_CTRL | Image statistics mode control | 0x0000 | 374 |
| 0x1076 | GREEN_1_SUM | Green 1 pixel sum | 0x0000 | 375 |
| 0x1078 | RED_SUM | Red pixel sum | 0x0000 | 375 |
| 0x107a | BLUE_SUM | Blue pixel sum | 0x0000 | 375 |
| 0x107c | GREEN_2_SUM | Green 2 pixel sum | 0x0000 | 375 |
| 0x107e | I_WIDTH | Current image width | 0x0000 | 376 |
| 0x1080 | I_HEIGHT | Current image height | 0x0000 | 377 |
| 0x1082 | STATUS_FLAGS | Status flags (read only) | 0x0000 | 378 |
| 0x1084 | CLK_GATE_DIS | Clock gate disable | 0x0000 | 379 |
| 0x1086 | CCIR_TIMING2 | CCIR interface timing 2 | 0x0000 | 381 |
| 0x1088 | CCIR_TIMING3 | CCIR interface timing 3 | 0x0010 | 382 |
| 0x108a | G1G2_DIAG_THRESH | Green 1/green 2 diagonal threshold | 0x0040 | 383 |
| 0x108c | BPA_D2_THRESH | BPA second derivative threshold | 0x0100 | 384 |
| 0x108e | SERIAL_CTRL | Serial control | 0x0000 | 385 |
| 0x1090 | INTP_CTRL_1 | Interpolation control 1 (demosaic) | 0x0188 | 387 |
| 0x1092 | INTP_CTRL_2 | Interpolation control 2 (demosaic) | 0x00c8 | 388 |
| 0x1094 | AV_OVAL_FACT | Anti-vignetting oval factor | 0x0100 | 389 |
| 0x1096 | AV_OS_GREEN1 | Anti-vignetting green 1 offset | 0x0000 | 391 |
| 0x1098 | AV_OS_RED | Anti-vignetting red offset | 0x0000 | 391 |
| 0x109a | AV_OS_BLUE | Anti-vignetting blue offset | 0x0000 | 391 |
| 0x109c | AV_OS_GREEN2 | Anti-vignetting green 2 offset | 0x0000 | 391 |

# By Mnemonic

**Registers by Mnemonic**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| A_FRAME_RATE | Actual frame rate | 0x0016 | 0x0096 | 154 |
| ABL_MAX_BIN | Auto black level maximum bin | 0x014c | 0x0003 | 226 |
| ABL_MAX_BLACK | Auto black level maximum black | 0x014e | 0x0010 | 227 |
| ABL_TARGET | Auto black level target | 0x014a | 0x0005 | 225 |
| ADC_CTRL | ADC control | 0x0809 | 0x01 | 277 |
| AE_DOE_FACTOR | AE deliberate overexposure factor | 0x0166 | 0x014e | 239 |
| AE_DOE_MARGIN | AE deliberate overexposure margin | 0x0168 | 0x0140 | 240 |
| AE_ETIME_DFLT | Auto exposure time default | 0x015c | 0x03e8 | 234 |
| AE_ETIME_MAX | Auto exposure time maximum | 0x015a | 0x4e20 | 233 |
| AE_ETIME_MIN | Auto exposure time minimum | 0x0158 | 0x0005 | 232 |
| AE_GAIN_DFLT | Auto exposure gain default | 0x0156 | 0x0200 | 231 |
| AE_GAIN_MAX | Auto exposure gain maximum | 0x0154 | 0x0500 | 230 |
| AE_GAIN_MIN | Auto exposure gain minimum | 0x0150 | 0x01c0 | 228 |
| AE_GAIN_MIN_P | Auto exposure gain minimum, preferred | 0x0152 | 0x0200 | 229 |
| AE_MARGIN | Auto exposure margin | 0x0164 | 0x0120 | 238 |
| AE_TARGET | Auto exposure target | 0x015e | 0x0040 | 235 |
| AE_TOL_ACQ | Auto exposure tolerance acquire | 0x0160 | 0x0118 | 236 |
| AE_TOL_MON | Auto exposure tolerance monitor | 0x0162 | 0x0118 | 237 |
| AF_CTRL1 | Auto functions control 1 | 0x0140 | 0x0013 | 220 |
| AF_CTRL2 | Auto functions control 2 | 0x0142 | 0x0001 | 221 |
| AF_STATUS | Auto functions status | 0x0144 | 0x0000 | 222 |
| APS_COEF_BLUE | Blue AWB gain | 0x1066 | 0x0080 | 368 |
| APS_COEF_GRN1 | Green 1 AWB gain | 0x1062 | 0x0080 | 368 |
| APS_COEF_GRN2 | Green 2 AWB gain | 0x1068 | 0x0080 | 368 |
| APS_COEF_RED | Red AWB gain | 0x1064 | 0x0080 | 368 |
| AV_CENTER_COL | Anti-vignetting, sensor center column | 0x106e | 0x0148 | 371 |
| AV_CENTER_ROW | Anti-vignetting, sensor center row | 0x1070 | 0x00f8 | 372 |
| AV_LEFT_TOP | Anti-vignetting, sensor first row and column | 0x106a | 0x0101 | 369 |
| AV_OS_BLUE | Anti-vignetting blue offset | 0x109a | 0x0000 | 391 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| AV_OS_GREEN1 | Anti-vignetting green 1 offset | 0x1096 | 0x0000 | 391 |
| AV_OS_GREEN2 | Anti-vignetting green 2 offset | 0x109c | 0x0000 | 391 |
| AV_OS_RED | Anti-vignetting red offset | 0x1098 | 0x0000 | 391 |
| AV_OVAL_FACT | Anti-vignetting oval factor | 0x1094 | 0x0100 | 389 |
| AV_RIGHT_BOT | Anti-vignetting, sensor last row and column | 0x106c | 0xa27a | 370 |
| AWB_BLUE_DFLT | AWB default blue/green ratio | 0x017a | 0x01e4 | 247 |
| AWB_BLUE_MAX | AWB maximum blue/green ratio | 0x0178 | 0x02a4 | 246 |
| AWB_BLUE_MIN | AWB minimum blue/green ratio | 0x0176 | 0x00c0 | 245 |
| AWB_RED_DFLT | AWB default red/green ratio | 0x0174 | 0x0134 | 244 |
| AWB_RED_MAX | AWB maximum red/green ratio | 0x0172 | 0x01a6 | 243 |
| AWB_RED_MIN | AWB minimum red/green ratio | 0x0170 | 0x00c0 | 242 |
| AWB_TOL_ACQ | Auto white balance tolerance acquire | 0x017c | 0x0110 | 248 |
| AWB_TOL_MON | Auto white balance tolerance monitor | 0x017e | 0x0120 | 249 |
| B_CR_MAX_MIN | Chrominance, Cr (or blue) maximum/minimum | 0x1016 | 0xff00 | 346 |
| BIAS_TRM | Bias trim | 0x0837 | 0x00 | 301 |
| BLUE_SUM | Blue pixel sum | 0x107a | 0x0000 | 375 |
| BPA_BADPIX_CNT | BPA bad pixel count (read only) | 0x101e | 0x0000 | 351 |
| BPA_D2_THRESH | BPA second derivative threshold | 0x108c | 0x0100 | 384 |
| BPA_OUTL_PED | BPA outlier, pedestal | 0x101c | 0x4008 | 350 |
| BPA_SF_GTHRESH | BPA scale factor, green filter threshold | 0x101a | 0x0220 | 349 |
| CC_COEF_00 | Color correction coefficient 00 | 0x1028 | 0x02f9 | 358 |
| CC_COEF_01 | Color correction coefficient 01 | 0x102a | 0x0f03 | 358 |
| CC_COEF_02 | Color correction coefficient 02 | 0x102c | 0x0f02 | 358 |
| CC_COEF_10 | Color correction coefficient 10 | 0x102e | 0x0f4f | 358 |
| CC_COEF_11 | Color correction coefficient 11 | 0x1030 | 0x025c | 358 |
| CC_COEF_12 | Color correction coefficient 12 | 0x1032 | 0x0f54 | 358 |
| CC_COEF_20 | Color correction coefficient 20 | 0x1034 | 0x0fe0 | 358 |
| CC_COEF_21 | Color correction coefficient 21 | 0x1036 | 0x0e4a | 358 |
| CC_COEF_22 | Color correction coefficient 22 | 0x1038 | 0x02d5 | 358 |
| CC_POST_OS_0 | Color correction post-offset 0 | 0x1040 | 0x0000 | 360 |
| CC_POST_OS_1 | Color correction post-offset 1 | 0x1042 | 0x0000 | 360 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| CC_POST_OS_2 | Color correction post-offset 2 | 0x1044 | 0x0000 | 360 |
| CC_PRE_OS_0 | Color correction pre-offset 0 | 0x103a | 0x01f8 | 360 |
| CC_PRE_OS_1 | Color correction pre-offset 1 | 0x103c | 0x01f8 | 360 |
| CC_PRE_OS_2 | Color correction pre-offset 2 | 0x103e | 0x01f8 | 360 |
| CCIR_TIMING | CCIR interface timing | 0x1010 | 0x0000 | 343 |
| CCIR_TIMING2 | CCIR interface timing 2 | 0x1086 | 0x0000 | 381 |
| CCIR_TIMING3 | CCIR interface timing 3 | 0x1088 | 0x0010 | 382 |
| CLK_DIV_S | Clock divisors, still mode | 0x0138 | 0x0000 | 216 |
| CLK_DIV_V | Clock divisors, video mode | 0x0118 | 0x0000 | 198 |
| CLK_FREQ | Input clock frequency | 0x0006 | 0x32c8 | 144 |
| CLK_GATE_DIS | Clock gate disable | 0x1084 | 0x0000 | 379 |
| CLK_PIXEL | Clocks per pixel | 0x080e | 0x02 | 282 |
| CMD_1 | Main command 1 | 0x1002 | 0x0000 | 335 |
| CMD_2 | Main command 2 (write 1's only) | 0x1004 | 0x0002 | 336 |
| CONFIG_1 | Image sensor configuration 1 | 0x081b | 0x0e | 294 |
| CONFIG_2 | Image sensor configuration 2 | 0x0827 | 0x00 | 298 |
| CONTROL | Camera control | 0x0002 | 0x0001 | 140 |
| CONTROL_1 | Image sensor control 1 | 0x081c | 0x24 | 295 |
| CPP_S | Clocks per pixel, still mode | 0x0128 | 0x0002 | 206 |
| CPP_V | Clocks per pixel, video mode | 0x0108 | 0x0002 | 188 |
| CSC_00_S | Color space conversion coefficient 00, still | 0x01a8 | 0x0026 | 258 |
| CSC_01_S | Color space conversion coefficient 01, still | 0x01aa | 0x004b | 258 |
| CSC_02_S | Color space conversion coefficient 02, still | 0x01ac | 0x000f | 258 |
| CSC_10_S | Color space conversion coefficient 10, still | 0x01ae | 0x01ed | 258 |
| CSC_11_S | Color space conversion coefficient 11, still | 0x01b0 | 0x01db | 258 |
| CSC_12_S | Color space conversion coefficient 12, still | 0x01b2 | 0x0038 | 258 |
| CSC_20_S | Color space conversion coefficient 20, still | 0x01b4 | 0x004f | 258 |
| CSC_21_S | Color space conversion coefficient 21, still | 0x01b6 | 0x01be | 258 |
| CSC_22_S | Color space conversion coefficient 22, still | 0x01b8 | 0x01f3 | 258 |
| CSC_00_V | Color space conversion coefficient 00, video | 0x0190 | 0x0026 | 256 |
| CSC_01_V | Color space conversion coefficient 01, video | 0x0192 | 0x004b | 256 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| CSC_02_V | Color space conversion coefficient 02, video | 0x0194 | 0x000f | 256 |
| CSC_10_V | Color space conversion coefficient 10, video | 0x0196 | 0x01ed | 256 |
| CSC_11_V | Color space conversion coefficient 11, video | 0x0198 | 0x01db | 256 |
| CSC_12_V | Color space conversion coefficient 12, video | 0x019a | 0x0038 | 256 |
| CSC_20_V | Color space conversion coefficient 20, video | 0x019c | 0x004f | 256 |
| CSC_21_V | Color space conversion coefficient 21, video | 0x019e | 0x01be | 256 |
| CSC_22_V | Color space conversion coefficient 22, video | 0x01a0 | 0x01f3 | 256 |
| CSC_COEF_00 | Color space conversion coefficient 00 | 0x1046 | 0x0026 | 363 |
| CSC_COEF_01 | Color space conversion coefficient 01 | 0x1048 | 0x004b | 363 |
| CSC_COEF_02 | Color space conversion coefficient 02 | 0x104a | 0x000f | 363 |
| CSC_COEF_10 | Color space conversion coefficient 10 | 0x104c | 0x01ed | 363 |
| CSC_COEF_11 | Color space conversion coefficient 11 | 0x104e | 0x01db | 363 |
| CSC_COEF_12 | Color space conversion coefficient 12 | 0x1050 | 0x0038 | 363 |
| CSC_COEF_20 | Color space conversion coefficient 20 | 0x1052 | 0x004f | 363 |
| CSC_COEF_21 | Color space conversion coefficient 21 | 0x1054 | 0x01be | 363 |
| CSC_COEF_22 | Color space conversion coefficient 22 | 0x1056 | 0x01f3 | 363 |
| CSC_OS_0 | Color space conversion offset 0 | 0x1058 | 0x0000 | 364 |
| CSC_OS_1 | Color space conversion offset 1 | 0x105a | 0x0080 | 364 |
| CSC_OS_2 | Color space conversion offset 2 | 0x105c | 0x0080 | 364 |
| CSC_OS0_S | Color space conversion offset 0, still | 0x01ba | 0x0000 | 259 |
| CSC_OS0_V | Color space conversion offset 0, video | 0x01a2 | 0x0000 | 257 |
| CSC_OS1_S | Color space conversion offset 1, still | 0x01bc | 0x0080 | 259 |
| CSC_OS1_V | Color space conversion offset 1, video | 0x01a4 | 0x0080 | 257 |
| CSC_OS2_S | Color space conversion offset 2, still | 0x01be | 0x0080 | 259 |
| CSC_OS2_V | Color space conversion offset 2, video | 0x01a6 | 0x0080 | 257 |
| CTL_CLK_DIV | Clock dividers for control and serial interfaces | 0x0082 | 0x4000 | 167 |
| DATA_GEN | Test data generator | 0x105e | 0x0000 | 365 |
| EOF_CODES | End of frame codes | 0x0148 | 0x0100 | 224 |
| EOF_CODES_W | End of frame codes working copy | 0x100e | 0x0100 | 342 |
| EREC_PGA | Even row, even column (green 1) PGA gain | 0x080f | 0x00 | 283 |
| EROC_PGA | Even row, odd column (red) PGA gain | 0x0810 | 0x00 | 284 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| ERROR | Error control | 0x0816 | 0x00 | 290 |
| EXP_ADJ | Exposure adjustment | 0x000e | 0x0000 | 150 |
| EXP_GR_E0 | Exposure, ground reference edge 0 | 0x08dd | 0x06 | 308 |
| EXP_GR_E1 | Exposure, ground reference edge 1 | 0x08dc | 0x10 | 307 |
| EXP_PRST_E0 | Exposure, preset edge 1 | 0x08fb | 0x06 | 326 |
| EXP_PRST_E1 | Exposure, preset edge 1 | 0x08fa | 0x02 | 325 |
| EXP_RST_E0 | Exposure, reset edge 1 | 0x08f1 | 0x03 | 317 |
| EXP_RST_E1 | Exposure, reset edge 1 | 0x08f0 | 0x10 | 316 |
| EXPOSURE | Exposure | 0x000c | 0x03e8 | 149 |
| FIRMWARE_REV | Current firmware revision | 0x0180 | 0x0152 | 250 |
| FLICK_CFG_1 | Flicker configuration 1 | 0x0182 | 0x2aeb | 251 |
| FLICK_CFG_2 | Flicker configuration 2 | 0x0184 | 0x0005 | 252 |
| FRAME_RATE | Frame rate | 0x0012 | 0x0096 | 152 |
| FWCOL | Window first column address | 0x080b | 0x01 | 279 |
| FWROW | Window first row address | 0x080a | 0x01 | 278 |
| G_CB_MAX_MIN | Chrominance, Cb (or green) maximum/minimum | 0x1014 | 0xff00 | 345 |
| G1G2_DIAG_THRESH | Green 1 / Green 2 diagonal threshold | 0x108a | 0x0040 | 383 |
| GR_POL | Ground reference polarity | 0x08df | 0xd3 | 310 |
| GREEN_1_SUM | Green 1 pixel sum | 0x1076 | 0x0000 | 375 |
| GREEN_2_SUM | Green 2 pixel sum | 0x107c | 0x0000 | 375 |
| GRR_CTRL | Ground reset reference control | 0x0828 | 0x00 | 299 |
| HBLANK | Horizontal blank | 0x0819 | 0x00 | 292 |
| HBLANK_S | Horizontal blanking period, still mode | 0x012a | 0x0000 | 207 |
| HBLANK_V | Horizontal blanking period, video mode | 0x010a | 0x0000 | 189 |
| HSYNC_PER | Horizontal synchronization period | 0x1060 | 0x0a8b | 366 |
| HYSNC_PER_S | HSYNC period, still mode | 0x0136 | 0x0545 | 215 |
| HYSNC_PER_V | HSYNC period, video mode | 0x0116 | 0x0a8b | 197 |
| I_CLK_DIV | Initial clock divider | 0x0080 | 0x0001 | 166 |
| I_HEIGHT | Current image height | 0x1080 | 0x0000 | 377 |
| I_WIDTH | Current image width | 0x107e | 0x0000 | 376 |
| ICTRL | Interface control | 0x0805 | 0x00 | 275 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| ID | Chip ID | 0x0000 | 0x0060 | 139 |
| IDENT | Image sensor ID | 0x0800 | 0x60 | 272 |
| ILLUM | Illumination | 0x0010 | 0x0000 | 151 |
| INTP_CTRL_1 | Interpolation control 1 (demosaic) | 0x1090 | 0x0188 | 387 |
| INTP_CTRL_2 | Interpolation control 2 (demosaic) | 0x1092 | 0x00c8 | 388 |
| IP_CLK_DIV | Clock dividers for image pipeline | 0x0086 | 0x0000 | 169 |
| IS_STATUS | Image sensor status | 0x0801 | 0x00 | 273 |
| LWCOL | Window last column address | 0x080d | 0xa2 | 281 |
| LWROW | Window last row address | 0x080c | 0x7a | 280 |
| MAX_SCLK | Maximum sensor clock | 0x018a | 0x1964 | 254 |
| MIN_MAX_F_S | Frame convergence rates, still mode | 0x012e | 0x0002 | 209 |
| MIN_MAX_F_V | Frame convergence rates, video mode | 0x010e | 0x0000 | 191 |
| NACC_BC_00 | NACC bright coefficients 00 | 0x0270 | 0x0235 | 266 |
| NACC_BC_01 | NACC bright coefficients 01 | 0x0272 | 0xff46 | 266 |
| NACC_BC_02 | NACC bright coefficients 02 | 0x0274 | 0xff85 | 266 |
| NACC_BC_10 | NACC bright coefficients 10 | 0x0276 | 0xff64 | 266 |
| NACC_BC_11 | NACC bright coefficients 11 | 0x0278 | 0x01fc | 266 |
| NACC_BC_12 | NACC bright coefficients 12 | 0x027a | 0xff9f | 266 |
| NACC_BC_20 | NACC bright coefficients 20 | 0x027c | 0x0008 | 266 |
| NACC_BC_21 | NACC bright coefficients 21 | 0x027e | 0xfe8d | 266 |
| NACC_BC_22 | NACC bright coefficients 22 | 0x0280 | 0x026b | 266 |
| NACC_DC_00 | NACC dark coefficients 00 | 0x0282 | 0x0048 | 266 |
| NACC_DC_01 | NACC dark coefficients 01 | 0x0284 | 0x010b | 266 |
| NACC_DC_02 | NACC dark coefficients 02 | 0x0286 | 0xffaa | 266 |
| NACC_DC_10 | NACC dark coefficients 10 | 0x0288 | 0x0048 | 266 |
| NACC_DC_11 | NACC dark coefficients 11 | 0x028a | 0x010b | 266 |
| NACC_DC_12 | NACC dark coefficients 12 | 0x028c | 0xffaa | 266 |
| NACC_DC_20 | NACC dark coefficients 20 | 0x028e | 0x0048 | 266 |
| NACC_DC_21 | NACC dark coefficients 21 | 0x0290 | 0x010b | 266 |
| NACC_DC_22 | NACC dark coefficients 22 | 0x0292 | 0xffaa | 266 |
| NACC_EGP_1 | Noise adaptive color correction, EGP 1 | 0x0250 | 0x05dc | 265 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| NACC_EGP_2 | Noise adaptive color correction, EGP 2 | 0x0254 | 0x0465 | 265 |
| NACC_EGP_3 | Noise adaptive color correction, EGP 3 | 0x0258 | 0x02ee | 265 |
| NACC_EGP_4 | Noise adaptive color correction, EGP 4 | 0x025c | 0x0177 | 265 |
| NACC_EGP_5 | Noise adaptive color correction, EGP 5 | 0x0260 | 0x0000 | 265 |
| NACC_EGP_6 | Noise adaptive color correction, EGP 6 | 0x0264 | 0x0000 | 265 |
| NACC_EGP_7 | Noise adaptive color correction, EGP 7 | 0x0268 | 0x0000 | 265 |
| NACC_EGP_8 | Noise adaptive color correction, EGP 8 | 0x026c | 0x0000 | 265 |
| NACC_SAT_1 | Noise adaptive color correction, saturation 1 | 0x0252 | 0x0000 | 265 |
| NACC_SAT_2 | Noise adaptive color correction, saturation 2 | 0x0256 | 0x0040 | 265 |
| NACC_SAT_3 | Noise adaptive color correction, saturation 3 | 0x025a | 0x0080 | 265 |
| NACC_SAT_4 | Noise adaptive color correction, saturation 4 | 0x025e | 0x00c0 | 265 |
| NACC_SAT_5 | Noise adaptive color correction, saturation 5 | 0x0262 | 0x0100 | 265 |
| NACC_SAT_6 | Noise adaptive color correction, saturation 6 | 0x0266 | 0x0000 | 265 |
| NACC_SAT_7 | Noise adaptive color correction, saturation 7 | 0x026a | 0x0000 | 265 |
| NACC_SAT_8 | Noise adaptive color correction, saturation 8 | 0x026e | 0x0000 | 265 |
| OREC_PGA | Odd row, even column (blue) PGA gain | 0x0811 | 0x00 | 285 |
| OROC_PGA | Odd row, odd column (green 2) PGA gain | 0x0812 | 0x00 | 286 |
| OUT_CTRL | Output control, simple control | 0x008e | 0x0000 | 173 |
| OUTPUT_CTRL | Output control working copy | 0x1008 | 0x9019 | 338 |
| OUTPUT_CTRL_S | Output control, still mode | 0x0130 | 0x8019 | 210 |
| OUTPUT_CTRL_V | Output control, video mode | 0x0110 | 0x9019 | 192 |
| OUTPUT_FORMAT | Output style | 0x000a | 0x0909 | 147 |
| OUTPUT_HGT_S | Output window height, still mode | 0x0026 | 0x0000 | 162 |
| OUTPUT_HGT_V | Output window height, video mode | 0x001e | 0x0000 | 158 |
| OUTPUT_WID_V | Output window width, video mode | 0x001c | 0x0000 | 157 |
| OUTPUT_WID_V | Output window width, still mode | 0x0024 | 0x0000 | 161 |
| PARALLEL_CTRL | Parallel output control working copy | 0x100a | 0x0000 | 340 |
| PARALLEL_CTRL_S | Parallel output control, still mode | 0x013a | 0x0000 | 217 |
| PARALLEL_CTRL_V | Parallel output control, video mode | 0x011a | 0x0003 | 199 |
| PLL_CTRL | PLL control | 0x0090 | 0x0024 | 174 |
| PLL_DIV_L | PLL divisors, large values | 0x0092 | 0x1d09 | 175 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| PLL_DIV_S | PLL divisors, small values | 0x0094 | 0x0200 | 176 |
| PRESET_POL | Preset polarity enable | 0x08fd | 0xd3 | 328 |
| PROC_CTRL_S | Processing control, still mode | 0x0132 | 0x0280 | 212 |
| PROC_CTRL_V | Processing control, video mode | 0x0112 | 0x0280 | 194 |
| PROCESS_CTRL | Processing control working copy | 0x1018 | 0x0280 | 347 |
| R_Y_MAX_MIN | Luminance, Y (or red) maximum/minimum | 0x1012 | 0xff00 | 344 |
| RED_SUM | Red pixel sum | 0x1078 | 0x0000 | 375 |
| RESET_POL | Reset polarity enable | 0x08f3 | 0xd3 | 319 |
| ROWEXP_H | Row exposure high | 0x0814 | 0x00 | 288 |
| ROWEXP_L | Row exposure low | 0x0813 | 0x54 | 287 |
| RPT_S | Row processing time, still mode | 0x0134 | 0x0546 | 214 |
| RPT_V | Row processing time, video mode | 0x0114 | 0x0546 | 196 |
| SEN_CLK_DIV | Sensor clock dividers | 0x0084 | 0x0000 | 168 |
| SEN_CTRL_S | Sensor control, still mode | 0x013c | 0x0000 | 218 |
| SEN_CTRL_V | Sensor control, video mode | 0x011c | 0x0000 | 200 |
| SENSOR_HGT_S | Sensor window height, still mode | 0x0022 | 0x0000 | 160 |
| SENSOR_HGT_V | Sensor window height, video mode | 0x001a | 0x0000 | 156 |
| SENSOR_WID_S | Sensor window width, still mode | 0x0020 | 0x0000 | 159 |
| SENSOR_WID_V | Sensor window width, video mode | 0x0018 | 0x0000 | 155 |
| SER_ADDR | Serial interface device address | 0x008a | 0x0053 | 171 |
| SER_PARM | Serial Interface parameters | 0x008c | 0x0000 | 172 |
| SERIAL_CTRL | Serial control | 0x108e | 0x0000 | 385 |
| SIZE | Image size and orientation | 0x0008 | 0x0605 | 145 |
| SMP_GR_E0 | Sample ground reference edge 0 | 0x08d9 | 0x0a | 305 |
| SMP_GR_E1 | Sample ground reference edge 1 | 0x08d8 | 0x10 | 304 |
| SMP_GR_E2 | Sample ground reference edge 2 | 0x08d7 | 0x00 | 303 |
| SMP_PRST_E0 | Sample, preset edge 0 | 0x08f7 | 0x0a | 323 |
| SMP_PRST_E1 | Sample, preset edge 1 | 0x08f6 | 0x02 | 322 |
| SMP_PRST_E2 | Sample, preset edge 2 | 0x08f5 | 0x00 | 321 |
| SMP_RST_E0 | Sample, reset edge 0 | 0x08ed | 0x07 | 314 |
| SMP_RST_E1 | Sample, reset edge 1 | 0x08ec | 0x10 | 313 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| SMP_RST_E2 | Sample, reset edge 2 | 0x08eb | 0x04 | 312 |
| SOF_CODE_W | Start of frame code working copy | 0x100c | 0x00ff | 341 |
| SOF_CODES | Start of frame codes | 0x0146 | 0xfeff | 223 |
| SROWEXP | Sub row exposure | 0x0815 | 0x31 | 289 |
| STAT_CAP_CTRL | Image statistics capture control | 0x1072 | 0x0021 | 373 |
| STAT_MODE_CTRL | Image statistics mode control | 0x1074 | 0x0000 | 374 |
| STATUS | Camera status | 0x0004 | 0x0004 | 142 |
| STATUS_FLAGS | Status flags (read only) | 0x1082 | 0x0000 | 378 |
| SZR_IN_H | Sizer input height | 0x1022 | 0x01e0 | 353 |
| SZR_IN_HGT_S | Sizer input height, still mode | 0x0122 | 0x01e0 | 203 |
| SZR_IN_HGT_V | Sizer input height, video mode | 0x0102 | 0x01e0 | 185 |
| SZR_IN_W | Sizer input width | 0x1020 | 0x0280 | 352 |
| SZR_IN_WID_S | Sizer input width, still mode | 0x0120 | 0x0280 | 202 |
| SZR_IN_WID_V | Sizer input width, video mode | 0x0100 | 0x0280 | 184 |
| SZR_OUT_H | Sizer output height | 0x1026 | 0x00f0 | 355 |
| SZR_OUT_HGT_S | Sizer output height, still mode | 0x0126 | 0x01e0 | 205 |
| SZR_OUT_HGT_V | Sizer output height, video mode | 0x0106 | 0x00f0 | 187 |
| SZR_OUT_W | Sizer output width | 0x1024 | 0x0140 | 354 |
| SZR_OUT_WID_S | Sizer output width, still mode | 0x0124 | 0x0280 | 204 |
| SZR_OUT_WID_V | Sizer output width, video mode | 0x0104 | 0x0140 | 186 |
| TM_COEF_00_S | Tonemap coefficient 00, still | 0x0202 | 0x0000 | 262 |
| TM_COEF_00_V | Tonemap coefficient 00, video | 0x01c0 | 0x0000 | 261 |
| TM_COEF_01_S | Tonemap coefficient 01, still | 0x0204 | 0x0017 | 262 |
| TM_COEF_01_V | Tonemap coefficient 01, video | 0x01c2 | 0x0017 | 261 |
| TM_COEF_02_S | Tonemap coefficient 02, still | 0x0206 | 0x0032 | 262 |
| TM_COEF_02_V | Tonemap coefficient 02, video | 0x01c4 | 0x0032 | 261 |
| TM_COEF_03_S | Tonemap coefficient 03, still | 0x0208 | 0x0046 | 262 |
| TM_COEF_03_V | Tonemap coefficient 03, video | 0x01c6 | 0x0046 | 261 |
| TM_COEF_04_S | Tonemap coefficient 04, still | 0x020a | 0x0056 | 262 |
| TM_COEF_04_V | Tonemap coefficient 04, video | 0x01c8 | 0x0056 | 261 |
| TM_COEF_05_S | Tonemap coefficient 05, still | 0x020c | 0x0064 | 262 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| TM_COEF_05_V | Tonemap coefficient 05, video | 0x01ca | 0x0064 | 261 |
| TM_COEF_06_S | Tonemap coefficient 06, still | 0x020e | 0x0071 | 262 |
| TM_COEF_06_V | Tonemap coefficient 06, video | 0x01cc | 0x0071 | 261 |
| TM_COEF_07_S | Tonemap coefficient 07, still | 0x0210 | 0x007c | 262 |
| TM_COEF_07_V | Tonemap coefficient 07, video | 0x01ce | 0x007c | 261 |
| TM_COEF_08_S | Tonemap coefficient 08, still | 0x0212 | 0x0086 | 262 |
| TM_COEF_08_V | Tonemap coefficient 08, video | 0x01d0 | 0x0086 | 261 |
| TM_COEF_09_S | Tonemap coefficient 09, still | 0x0214 | 0x0099 | 262 |
| TM_COEF_09_V | Tonemap coefficient 09, video | 0x01d2 | 0x0099 | 261 |
| TM_COEF_10_S | Tonemap coefficient 10, still | 0x0216 | 0x00a9 | 262 |
| TM_COEF_10_V | Tonemap coefficient 10, video | 0x01d4 | 0x00a9 | 261 |
| TM_COEF_11_S | Tonemap coefficient 11, still | 0x0218 | 0x00b8 | 262 |
| TM_COEF_11_V | Tonemap coefficient 11, video | 0x01d6 | 0x00b8 | 261 |
| TM_COEF_12_S | Tonemap coefficient 12, still | 0x021a | 0x00c6 | 262 |
| TM_COEF_12_V | Tonemap coefficient 12, video | 0x01d8 | 0x00c6 | 261 |
| TM_COEF_13_S | Tonemap coefficient 13, still | 0x021c | 0x00df | 262 |
| TM_COEF_13_V | Tonemap coefficient 13, video | 0x01da | 0x00df | 261 |
| TM_COEF_14_S | Tonemap coefficient 14, still | 0x021e | 0x00f5 | 262 |
| TM_COEF_14_V | Tonemap coefficient 14, video | 0x01dc | 0x00f5 | 261 |
| TM_COEF_15_S | Tonemap coefficient 15, still | 0x0220 | 0x0109 | 262 |
| TM_COEF_15_V | Tonemap coefficient 15, video | 0x01de | 0x0109 | 261 |
| TM_COEF_16_S | Tonemap coefficient 16, still | 0x0222 | 0x011b | 262 |
| TM_COEF_16_V | Tonemap coefficient 16, video | 0x01e0 | 0x011b | 261 |
| TM_COEF_17_S | Tonemap coefficient 17, still | 0x0224 | 0x013d | 262 |
| TM_COEF_17_V | Tonemap coefficient 17, video | 0x01e2 | 0x013d | 261 |
| TM_COEF_18_S | Tonemap coefficient 18, still | 0x0226 | 0x015a | 262 |
| TM_COEF_18_V | Tonemap coefficient 18, video | 0x01e4 | 0x015a | 261 |
| TM_COEF_19_S | Tonemap coefficient 19, still | 0x0228 | 0x0175 | 262 |
| TM_COEF_19_V | Tonemap coefficient 19, video | 0x01e6 | 0x0175 | 261 |
| TM_COEF_20_S | Tonemap coefficient 20, still | 0x022a | 0x018d | 262 |
| TM_COEF_20_V | Tonemap coefficient 20, video | 0x01e8 | 0x018d | 261 |

**Registers by Mnemonic (continued)**

| Mnemonic | Description | Address | Default | Page |
|---|---|---|---|---|
| TM_COEF_21_S | Tonemap coefficient 21, still | 0x022c | 0x01ba | 262 |
| TM_COEF_21_V | Tonemap coefficient 21, video | 0x01ea | 0x01ba | 261 |
| TM_COEF_22_S | Tonemap coefficient 22, still | 0x022e | 0x01e1 | 262 |
| TM_COEF_22_V | Tonemap coefficient 22, video | 0x01ec | 0x01e1 | 261 |
| TM_COEF_23_S | Tonemap coefficient 23, still | 0x0230 | 0x0205 | 262 |
| TM_COEF_23_V | Tonemap coefficient 23, video | 0x01ee | 0x0205 | 261 |
| TM_COEF_24_S | Tonemap coefficient 24, still | 0x0232 | 0x0225 | 262 |
| TM_COEF_24_V | Tonemap coefficient 24, video | 0x01f0 | 0x0225 | 261 |
| TM_COEF_25_S | Tonemap coefficient 25, still | 0x0234 | 0x0261 | 262 |
| TM_COEF_25_V | Tonemap coefficient 25, video | 0x01f2 | 0x0261 | 261 |
| TM_COEF_26_S | Tonemap coefficient 26, still | 0x0236 | 0x0295 | 262 |
| TM_COEF_26_V | Tonemap coefficient 26, video | 0x01f4 | 0x0295 | 261 |
| TM_COEF_27_S | Tonemap coefficient 27, still | 0x0238 | 0x02c5 | 262 |
| TM_COEF_27_V | Tonemap coefficient 27, video | 0x01f6 | 0x02c5 | 261 |
| TM_COEF_28_S | Tonemap coefficient 28, still | 0x023a | 0x02f1 | 262 |
| TM_COEF_28_V | Tonemap coefficient 28, video | 0x01f8 | 0x02f1 | 261 |
| TM_COEF_29_S | Tonemap coefficient 29, still | 0x023c | 0x033f | 262 |
| TM_COEF_29_V | Tonemap coefficient 29, video | 0x01fa | 0x033f | 261 |
| TM_COEF_30_S | Tonemap coefficient 30, still | 0x023e | 0x0385 | 262 |
| TM_COEF_30_V | Tonemap coefficient 30, video | 0x01fc | 0x0385 | 261 |
| TM_COEF_31_S | Tonemap coefficient 31, still | 0x0240 | 0x03c5 | 262 |
| TM_COEF_31_V | Tonemap coefficient 31, video | 0x01fe | 0x03c5 | 261 |
| TM_COEF_32_S | Tonemap coefficient 32, still | 0x0242 | 0x0400 | 262 |
| TM_COEF_32_V | Tonemap coefficient 32, video | 0x0200 | 0x0400 | 261 |
| TST_MODE | Latched test mode | 0x0088 | 0x0000 | 170 |
| VBLANK | Vertical blank | 0x081a | 0x00 | 293 |
| VBLANK_S | Vertical blanking period, still mode | 0x012c | 0x0000 | 208 |
| VBLANK_V | Vertical blanking period, video mode | 0x010c | 0x0000 | 190 |

# C
# RAM Locations

# Image Pipeline RAMs

There are eight separate RAM locations used by the image pipeline.

**RAM addresses**

| Address | RAM Block | Description |
|---|---|---|
| 0x1400 to 0x1440 | TM_ALL | Tonemap for all colors, if data is written to these locations, the data in the individual color tonemaps is overwritten. If data is read from these locations, the module will return the values in the RED table. |
| 0x1442 to 0x147e | Reserved | Reserved |
| 0x1480 to 0x14c0 | TM_RED | Tonemap for the red pixels |
| 0x14c2 to 0x14fe | Reserved | Reserved |
| 0x1500 to 0x1540 | TM_GREEN | Tonemap for the green 1 and green 2 pixels |
| 0x1542 to 0x157e | Reserved | Reserved |
| 0x1580 to 0x15c0 | TM_BLUE | Tonemap for the blue pixels |
| 0x15c2 to 0x15fe | Reserved | Reserved |
| 0x1600 to 0x16be | STATS | Image statistics used by the auto functions<br>Histogram or flicker statistics |
| 0x16c0 to 0x17fe | Reserved | Reserved |
| 0x1800 to 0x183e | AV_RED | Anti-vignetting table for the red pixels |
| 0x1840 to 0x187e | AV_GREEN | Anti-vignetting table for the green pixels |
| 0x1880 to 0x18be | AV_BLUE | Anti-vignetting table for the blue pixels |

## TM_ALL

**Tonemap, all colors**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1400 | TONEMAP_ALL_0 | 0x0000 | 0x28 | 0x00 |
| 0x1402 | TONEMAP_ALL_1 | 0x0017 | 0x28 | 0x02 |
| 0x1404 | TONEMAP_ALL_2 | 0x0032 | 0x28 | 0x04 |
| 0x1406 | TONEMAP_ALL_3 | 0x0046 | 0x28 | 0x06 |
| 0x1408 | TONEMAP_ALL_4 | 0x0056 | 0x28 | 0x08 |
| 0x140a | TONEMAP_ALL_5 | 0x0064 | 0x28 | 0x0a |
| 0x140c | TONEMAP_ALL_6 | 0x0071 | 0x28 | 0x0c |
| 0x140e | TONEMAP_ALL_7 | 0x007c | 0x28 | 0x0e |
| 0x1410 | TONEMAP_ALL_8 | 0x0086 | 0x28 | 0x10 |
| 0x1412 | TONEMAP_ALL_9 | 0x0099 | 0x28 | 0x12 |
| 0x1414 | TONEMAP_ALL_10 | 0x00a9 | 0x28 | 0x14 |
| 0x1416 | TONEMAP_ALL_11 | 0x00b8 | 0x28 | 0x16 |
| 0x1418 | TONEMAP_ALL_12 | 0x00c6 | 0x28 | 0x18 |
| 0x141a | TONEMAP_ALL_13 | 0x00df | 0x28 | 0x1a |
| 0x141c | TONEMAP_ALL_14 | 0x00f5 | 0x28 | 0x1c |
| 0x141e | TONEMAP_ALL_15 | 0x0109 | 0x28 | 0x1e |
| 0x1420 | TONEMAP_ALL_16 | 0x011b | 0x28 | 0x20 |
| 0x1422 | TONEMAP_ALL_17 | 0x013d | 0x28 | 0x22 |
| 0x1424 | TONEMAP_ALL_18 | 0x015a | 0x28 | 0x24 |
| 0x1426 | TONEMAP_ALL_19 | 0x0175 | 0x28 | 0x26 |
| 0x1428 | TONEMAP_ALL_20 | 0x018d | 0x28 | 0x28 |
| 0x142a | TONEMAP_ALL_21 | 0x01ba | 0x28 | 0x2a |
| 0x142c | TONEMAP_ALL_22 | 0x01e1 | 0x28 | 0x2c |
| 0x142e | TONEMAP_ALL_23 | 0x0205 | 0x28 | 0x2e |
| 0x1430 | TONEMAP_ALL_24 | 0x0225 | 0x28 | 0x30 |
| 0x1432 | TONEMAP_ALL_25 | 0x0261 | 0x28 | 0x32 |
| 0x1434 | TONEMAP_ALL_26 | 0x0295 | 0x28 | 0x34 |
| 0x1436 | TONEMAP_ALL_27 | 0x02c5 | 0x28 | 0x36 |
| 0x1438 | TONEMAP_ALL_28 | 0x02f1 | 0x28 | 0x38 |
| 0x143a | TONEMAP_ALL_29 | 0x033f | 0x28 | 0x3a |
| 0x143c | TONEMAP_ALL_30 | 0x0385 | 0x28 | 0x3c |
| 0x143e | TONEMAP_ALL_31 | 0x03c5 | 0x28 | 0x3e |
| 0x1440 | TONEMAP_ALL_32 | 0x0400 | 0x28 | 0x40 |

## TM_RED

**Tonemap, red pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1480 | TONEMAP_RED_0 | 0x0000 | 0x29 | 0x00 |
| 0x1482 | TONEMAP_RED_1 | 0x0017 | 0x29 | 0x02 |
| 0x1484 | TONEMAP_RED_2 | 0x0032 | 0x29 | 0x04 |
| 0x1486 | TONEMAP_RED_3 | 0x0046 | 0x29 | 0x06 |
| 0x1488 | TONEMAP_RED_4 | 0x0056 | 0x29 | 0x08 |
| 0x148a | TONEMAP_RED_5 | 0x0064 | 0x29 | 0x0a |
| 0x148c | TONEMAP_RED_6 | 0x0071 | 0x29 | 0x0c |
| 0x148e | TONEMAP_RED_7 | 0x007c | 0x29 | 0x0e |
| 0x1490 | TONEMAP_RED_8 | 0x0086 | 0x29 | 0x10 |
| 0x1492 | TONEMAP_RED_9 | 0x0099 | 0x29 | 0x12 |
| 0x1494 | TONEMAP_RED_10 | 0x00a9 | 0x29 | 0x14 |
| 0x1496 | TONEMAP_RED_11 | 0x00b8 | 0x29 | 0x16 |
| 0x1498 | TONEMAP_RED_12 | 0x00c6 | 0x29 | 0x18 |
| 0x149a | TONEMAP_RED_13 | 0x00df | 0x29 | 0x1a |
| 0x149c | TONEMAP_RED_14 | 0x00f5 | 0x29 | 0x1c |
| 0x149e | TONEMAP_RED_15 | 0x0109 | 0x29 | 0x1e |
| 0x14a0 | TONEMAP_RED_16 | 0x011b | 0x29 | 0x20 |
| 0x14a2 | TONEMAP_RED_17 | 0x013d | 0x29 | 0x22 |
| 0x14a4 | TONEMAP_RED_18 | 0x015a | 0x29 | 0x24 |
| 0x14a6 | TONEMAP_RED_19 | 0x0175 | 0x29 | 0x26 |
| 0x14a8 | TONEMAP_RED_20 | 0x018d | 0x29 | 0x28 |
| 0x14aa | TONEMAP_RED_21 | 0x01ba | 0x29 | 0x2a |
| 0x14ac | TONEMAP_RED_22 | 0x01e1 | 0x29 | 0x2c |
| 0x14ae | TONEMAP_RED_23 | 0x0205 | 0x29 | 0x2e |
| 0x14b0 | TONEMAP_RED_24 | 0x0225 | 0x29 | 0x30 |
| 0x14b2 | TONEMAP_RED_25 | 0x0261 | 0x29 | 0x32 |
| 0x14b4 | TONEMAP_RED_26 | 0x0295 | 0x29 | 0x34 |
| 0x14b6 | TONEMAP_RED_27 | 0x02c5 | 0x29 | 0x36 |
| 0x14b8 | TONEMAP_RED_28 | 0x02f1 | 0x29 | 0x38 |
| 0x14ba | TONEMAP_RED_29 | 0x033f | 0x29 | 0x3a |
| 0x14bc | TONEMAP_RED_30 | 0x0385 | 0x29 | 0x3c |
| 0x14be | TONEMAP_RED_31 | 0x03c5 | 0x29 | 0x3e |
| 0x14c0 | TONEMAP_RED_32 | 0x0400 | 0x29 | 0x40 |

## TM_GREEN

**Tonemap, green pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1500 | TONEMAP_GREEN_0 | 0x0000 | 0x2a | 0x00 |
| 0x1502 | TONEMAP_GREEN_1 | 0x0017 | 0x2a | 0x02 |
| 0x1504 | TONEMAP_GREEN_2 | 0x0032 | 0x2a | 0x04 |
| 0x1506 | TONEMAP_GREEN_3 | 0x0046 | 0x2a | 0x06 |
| 0x1508 | TONEMAP_GREEN_4 | 0x0056 | 0x2a | 0x08 |
| 0x150a | TONEMAP_GREEN_5 | 0x0064 | 0x2a | 0x0a |
| 0x150c | TONEMAP_GREEN_6 | 0x0071 | 0x2a | 0x0c |
| 0x150e | TONEMAP_GREEN_7 | 0x007c | 0x2a | 0x0e |
| 0x1510 | TONEMAP_GREEN_8 | 0x0086 | 0x2a | 0x10 |
| 0x1512 | TONEMAP_GREEN_9 | 0x0099 | 0x2a | 0x12 |
| 0x1514 | TONEMAP_GREEN_10 | 0x00a9 | 0x2a | 0x14 |
| 0x1516 | TONEMAP_GREEN_11 | 0x00b8 | 0x2a | 0x16 |
| 0x1518 | TONEMAP_GREEN_12 | 0x00c6 | 0x2a | 0x18 |
| 0x151a | TONEMAP_GREEN_13 | 0x00df | 0x2a | 0x1a |
| 0x151c | TONEMAP_GREEN_14 | 0x00f5 | 0x2a | 0x1c |
| 0x151e | TONEMAP_GREEN_15 | 0x0109 | 0x2a | 0x1e |
| 0x1520 | TONEMAP_GREEN_16 | 0x011b | 0x2a | 0x20 |
| 0x1522 | TONEMAP_GREEN_17 | 0x013d | 0x2a | 0x22 |
| 0x1524 | TONEMAP_GREEN_18 | 0x015a | 0x2a | 0x24 |
| 0x1526 | TONEMAP_GREEN_19 | 0x0175 | 0x2a | 0x26 |
| 0x1528 | TONEMAP_GREEN_20 | 0x018d | 0x2a | 0x28 |
| 0x152a | TONEMAP_GREEN_21 | 0x01ba | 0x2a | 0x2a |
| 0x152c | TONEMAP_GREEN_22 | 0x01e1 | 0x2a | 0x2c |
| 0x152e | TONEMAP_GREEN_23 | 0x0205 | 0x2a | 0x2e |
| 0x1530 | TONEMAP_GREEN_24 | 0x0225 | 0x2a | 0x30 |
| 0x1532 | TONEMAP_GREEN_25 | 0x0261 | 0x2a | 0x32 |
| 0x1534 | TONEMAP_GREEN_26 | 0x0295 | 0x2a | 0x34 |
| 0x1536 | TONEMAP_GREEN_27 | 0x02c5 | 0x2a | 0x36 |
| 0x1538 | TONEMAP_GREEN_28 | 0x02f1 | 0x2a | 0x38 |
| 0x153a | TONEMAP_GREEN_29 | 0x033f | 0x2a | 0x3a |
| 0x153c | TONEMAP_GREEN_30 | 0x0385 | 0x2a | 0x3c |
| 0x153e | TONEMAP_GREEN_31 | 0x03c5 | 0x2a | 0x3e |
| 0x1540 | TONEMAP_GREEN_32 | 0x0400 | 0x2a | 0x40 |

## TM_BLUE

**Tonemap, blue pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1580 | TONEMAP_BLUE_0 | 0x0000 | 0x2b | 0x00 |
| 0x1582 | TONEMAP_BLUE_1 | 0x0017 | 0x2b | 0x02 |
| 0x1584 | TONEMAP_BLUE_2 | 0x0032 | 0x2b | 0x04 |
| 0x1586 | TONEMAP_BLUE_3 | 0x0046 | 0x2b | 0x06 |
| 0x1588 | TONEMAP_BLUE_4 | 0x0056 | 0x2b | 0x08 |
| 0x158a | TONEMAP_BLUE_5 | 0x0064 | 0x2b | 0x0a |
| 0x158c | TONEMAP_BLUE_6 | 0x0071 | 0x2b | 0x0c |
| 0x158e | TONEMAP_BLUE_7 | 0x007c | 0x2b | 0x0e |
| 0x1590 | TONEMAP_BLUE_8 | 0x0086 | 0x2b | 0x10 |
| 0x1592 | TONEMAP_BLUE_9 | 0x0099 | 0x2b | 0x12 |
| 0x1594 | TONEMAP_BLUE_10 | 0x00a9 | 0x2b | 0x14 |
| 0x1596 | TONEMAP_BLUE_11 | 0x00b8 | 0x2b | 0x16 |
| 0x1598 | TONEMAP_BLUE_12 | 0x00c6 | 0x2b | 0x18 |
| 0x159a | TONEMAP_BLUE_13 | 0x00df | 0x2b | 0x1a |
| 0x159c | TONEMAP_BLUE_14 | 0x00f5 | 0x2b | 0x1c |
| 0x159e | TONEMAP_BLUE_15 | 0x0109 | 0x2b | 0x1e |
| 0x15a0 | TONEMAP_BLUE_16 | 0x011b | 0x2b | 0x20 |
| 0x15a2 | TONEMAP_BLUE_17 | 0x013d | 0x2b | 0x22 |
| 0x15a4 | TONEMAP_BLUE_18 | 0x015a | 0x2b | 0x24 |
| 0x15a6 | TONEMAP_BLUE_19 | 0x0175 | 0x2b | 0x26 |
| 0x15a8 | TONEMAP_BLUE_20 | 0x018d | 0x2b | 0x28 |
| 0x15aa | TONEMAP_BLUE_21 | 0x01ba | 0x2b | 0x2a |
| 0x15ac | TONEMAP_BLUE_22 | 0x01e1 | 0x2b | 0x2c |
| 0x15ae | TONEMAP_BLUE_23 | 0x0205 | 0x2b | 0x2e |
| 0x15b0 | TONEMAP_BLUE_24 | 0x0225 | 0x2b | 0x30 |
| 0x15b2 | TONEMAP_BLUE_25 | 0x0261 | 0x2b | 0x32 |
| 0x15b4 | TONEMAP_BLUE_26 | 0x0295 | 0x2b | 0x34 |
| 0x15b6 | TONEMAP_BLUE_27 | 0x02c5 | 0x2b | 0x36 |
| 0x15b8 | TONEMAP_BLUE_28 | 0x02f1 | 0x2b | 0x38 |
| 0x15ba | TONEMAP_BLUE_29 | 0x033f | 0x2b | 0x3a |
| 0x15bc | TONEMAP_BLUE_30 | 0x0385 | 0x2b | 0x3c |
| 0x15be | TONEMAP_BLUE_31 | 0x03c5 | 0x2b | 0x3e |
| 0x15c0 | TONEMAP_BLUE_32 | 0x0400 | 0x2b | 0x40 |

# Image Statistics

The RAM locations 0x1600 to 0x16de change function depending on the mode of the camera module. The two major modes are histogram or flicker statistics.

## Green 1 Histogram

**Histogram, green 1 image**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1600 | GREEN_1_HISTO_BIN_0 | 0x0000 | 0x2c | 0x00 |
| 0x1602 | GREEN_1_HISTO_BIN_1 | 0x0000 | 0x2c | 0x02 |
| 0x1604 | GREEN_1_HISTO_BIN_2 | 0x0000 | 0x2c | 0x04 |
| 0x1606 | GREEN_1_HISTO_BIN_3 | 0x0000 | 0x2c | 0x06 |
| 0x1608 | GREEN_1_HISTO_BIN_4 | 0x0000 | 0x2c | 0x08 |
| 0x160a | GREEN_1_HISTO_BIN_5 | 0x0000 | 0x2c | 0x0a |
| 0x160c | GREEN_1_HISTO_BIN_6 | 0x0000 | 0x2c | 0x0c |
| 0x160e | GREEN_1_HISTO_BIN_7 | 0x0000 | 0x2c | 0x0e |
| 0x1610 | GREEN_1_HISTO_BIN_8 | 0x0000 | 0x2c | 0x10 |
| 0x1612 | GREEN_1_HISTO_BIN_9 | 0x0000 | 0x2c | 0x12 |
| 0x1614 | GREEN_1_HISTO_BIN_10 | 0x0000 | 0x2c | 0x14 |
| 0x1616 | GREEN_1_HISTO_BIN_11 | 0x0000 | 0x2c | 0x16 |
| 0x1618 | GREEN_1_HISTO_BIN_12 | 0x0000 | 0x2c | 0x18 |
| 0x161a | GREEN_1_HISTO_BIN_13 | 0x0000 | 0x2c | 0x1a |
| 0x161c | GREEN_1_HISTO_BIN_14 | 0x0000 | 0x2c | 0x1c |
| 0x161e | GREEN_1_HISTO_BIN_15 | 0x0000 | 0x2c | 0x1e |
| 0x1620 | GREEN_1_HISTO_BIN_16 | 0x0000 | 0x2c | 0x20 |
| 0x1622 | GREEN_1_HISTO_BIN_17 | 0x0000 | 0x2c | 0x22 |
| 0x1624 | GREEN_1_HISTO_BIN_18 | 0x0000 | 0x2c | 0x24 |
| 0x1626 | GREEN_1_HISTO_BIN_19 | 0x0000 | 0x2c | 0x26 |

## Red Histogram

**Histogram, red image**

| Full Address | Mnemonic | Default Value | Address Block | Address Offset |
|---|---|---|---|---|
| 0x1628 | RED_HISTO_BIN_0 | 0x0000 | 0x2c | 0x28 |
| 0x162a | RED_HISTO_BIN_1 | 0x0000 | 0x2c | 0x2a |
| 0x162c | RED_HISTO_BIN_2 | 0x0000 | 0x2c | 0x2c |
| 0x162e | RED_HISTO_BIN_3 | 0x0000 | 0x2c | 0x2e |
| 0x1630 | RED_HISTO_BIN_4 | 0x0000 | 0x2c | 0x30 |
| 0x1632 | RED_HISTO_BIN_5 | 0x0000 | 0x2c | 0x32 |
| 0x1634 | RED_HISTO_BIN_6 | 0x0000 | 0x2c | 0x34 |
| 0x1636 | RED_HISTO_BIN_7 | 0x0000 | 0x2c | 0x36 |
| 0x1638 | RED_HISTO_BIN_8 | 0x0000 | 0x2c | 0x38 |
| 0x163a | RED_HISTO_BIN_9 | 0x0000 | 0x2c | 0x3a |
| 0x163c | RED_HISTO_BIN_10 | 0x0000 | 0x2c | 0x3c |
| 0x163e | RED_HISTO_BIN_11 | 0x0000 | 0x2c | 0x3e |
| 0x1640 | RED_HISTO_BIN_12 | 0x0000 | 0x2c | 0x40 |
| 0x1642 | RED_HISTO_BIN_13 | 0x0000 | 0x2c | 0x42 |
| 0x1644 | RED_HISTO_BIN_14 | 0x0000 | 0x2c | 0x44 |
| 0x1646 | RED_HISTO_BIN_15 | 0x0000 | 0x2c | 0x46 |
| 0x1648 | RED_HISTO_BIN_16 | 0x0000 | 0x2c | 0x48 |
| 0x164a | RED_HISTO_BIN_17 | 0x0000 | 0x2c | 0x4a |
| 0x164c | RED_HISTO_BIN_18 | 0x0000 | 0x2c | 0x4c |
| 0x164e | RED_HISTO_BIN_19 | 0x0000 | 0x2c | 0x4e |

## Blue Histogram

**Histogram, blue image**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1650 | BLUE_HISTO_BIN_0 | 0x0000 | 0x2c | 0x50 |
| 0x1652 | BLUE_HISTO_BIN_1 | 0x0000 | 0x2c | 0x52 |
| 0x1654 | BLUE_HISTO_BIN_2 | 0x0000 | 0x2c | 0x54 |
| 0x1656 | BLUE_HISTO_BIN_3 | 0x0000 | 0x2c | 0x56 |
| 0x1658 | BLUE_HISTO_BIN_4 | 0x0000 | 0x2c | 0x58 |
| 0x165a | BLUE_HISTO_BIN_5 | 0x0000 | 0x2c | 0x5a |
| 0x165c | BLUE_HISTO_BIN_6 | 0x0000 | 0x2c | 0x5c |
| 0x165e | BLUE_HISTO_BIN_7 | 0x0000 | 0x2c | 0x5e |
| 0x1660 | BLUE_HISTO_BIN_8 | 0x0000 | 0x2c | 0x60 |
| 0x1662 | BLUE_HISTO_BIN_9 | 0x0000 | 0x2c | 0x62 |
| 0x1664 | BLUE_HISTO_BIN_10 | 0x0000 | 0x2c | 0x64 |
| 0x1666 | BLUE_HISTO_BIN_11 | 0x0000 | 0x2c | 0x66 |
| 0x1668 | BLUE_HISTO_BIN_12 | 0x0000 | 0x2c | 0x68 |
| 0x166a | BLUE_HISTO_BIN_13 | 0x0000 | 0x2c | 0x6a |
| 0x166c | BLUE_HISTO_BIN_14 | 0x0000 | 0x2c | 0x6c |
| 0x166e | BLUE_HISTO_BIN_15 | 0x0000 | 0x2c | 0x6e |
| 0x1670 | BLUE_HISTO_BIN_16 | 0x0000 | 0x2c | 0x70 |
| 0x1672 | BLUE_HISTO_BIN_17 | 0x0000 | 0x2c | 0x72 |
| 0x1674 | BLUE_HISTO_BIN_18 | 0x0000 | 0x2c | 0x74 |
| 0x1676 | BLUE_HISTO_BIN_19 | 0x0000 | 0x2c | 0x76 |

## Green 2 Histogram

**Histogram, green 2 image**

| Full | | Default | Address | |
|---|---|---|---|---|
| Address | Mnemonic | Value | Block | Offset |
| 0x1678 | GREEN_2_HISTO_BIN_0 | 0x0000 | 0x2c | 0x78 |
| 0x167a | GREEN_2_HISTO_BIN_1 | 0x0000 | 0x2c | 0x7a |
| 0x167c | GREEN_2_HISTO_BIN_2 | 0x0000 | 0x2c | 0x7c |
| 0x167e | GREEN_2_HISTO_BIN_3 | 0x0000 | 0x2c | 0x7e |
| 0x1680 | GREEN_2_HISTO_BIN_4 | 0x0000 | 0x2d | 0x00 |
| 0x1682 | GREEN_2_HISTO_BIN_5 | 0x0000 | 0x2d | 0x02 |
| 0x1684 | GREEN_2_HISTO_BIN_6 | 0x0000 | 0x2d | 0x04 |
| 0x1686 | GREEN_2_HISTO_BIN_7 | 0x0000 | 0x2d | 0x06 |
| 0x1688 | GREEN_2_HISTO_BIN_8 | 0x0000 | 0x2d | 0x08 |
| 0x168a | GREEN_2_HISTO_BIN_9 | 0x0000 | 0x2d | 0x0a |
| 0x168c | GREEN_2_HISTO_BIN_10 | 0x0000 | 0x2d | 0x0c |
| 0x168e | GREEN_2_HISTO_BIN_11 | 0x0000 | 0x2d | 0x0e |
| 0x1690 | GREEN_2_HISTO_BIN_12 | 0x0000 | 0x2d | 0x10 |
| 0x1692 | GREEN_2_HISTO_BIN_13 | 0x0000 | 0x2d | 0x12 |
| 0x1694 | GREEN_2_HISTO_BIN_14 | 0x0000 | 0x2d | 0x14 |
| 0x1696 | GREEN_2_HISTO_BIN_15 | 0x0000 | 0x2d | 0x16 |
| 0x1698 | GREEN_2_HISTO_BIN_16 | 0x0000 | 0x2d | 0x18 |
| 0x169a | GREEN_2_HISTO_BIN_17 | 0x0000 | 0x2d | 0x1a |
| 0x169c | GREEN_2_HISTO_BIN_18 | 0x0000 | 0x2d | 0x1c |
| 0x169e | GREEN_2_HISTO_BIN_19 | 0x0000 | 0x2d | 0x1e |

# Flicker Statistics

The addresses for the row sums for each frame are dependent on the number of rows in the frame and the flicker skip count (see “Flicker Statistics" on page 47 for additional information).

## Example 1

Flicker statistics, CIF window (296 rows), flicker skip count from one to five, resulting in 48 row sums per frame.

**Flicker statistics: row sums, 48 rows/frame, N = flicker skip count**

| | Frame 1 | | | Frame 2 | | |
|---|---|---|---|---|---|---|
| **Description** | **Address** | **Block** | **Offset** | **Address** | **Block** | **Offset** |
| Row 1 | 0x1600 | 0x2c | 0x00 | 0x1660 | 0x2c | 0x60 |
| Row 1 + (N+1) | 0x1602 | 0x2c | 0x02 | 0x1662 | 0x2c | 0x62 |
| Row 1 + 2 x (N+1) | 0x1604 | 0x2c | 0x04 | 0x1664 | 0x2c | 0x64 |
| Row 1 + 3 x (N+1) | 0x1606 | 0x2c | 0x06 | 0x1666 | 0x2c | 0x66 |
| Row 1 + 4 x (N+1) | 0x1608 | 0x2c | 0x08 | 0x1668 | 0x2c | 0x68 |
| Row 1 + 5 x (N+1) | 0x160a | 0x2c | 0x0a | 0x166a | 0x2c | 0x6a |
| Row 1 + 6 x (N+1) | 0x160c | 0x2c | 0x0c | 0x166c | 0x2c | 0x6c |
| Row 1 + 7 x (N+1) | 0x160e | 0x2c | 0x0e | 0x166e | 0x2c | 0x6e |
| Row 1 + 8 x (N+1) | 0x1610 | 0x2c | 0x10 | 0x1670 | 0x2c | 0x70 |
| Row 1 + 9 x (N+1) | 0x1612 | 0x2c | 0x12 | 0x1672 | 0x2c | 0x72 |
| Row 1 + 10 x (N+1) | 0x1614 | 0x2c | 0x14 | 0x1674 | 0x2c | 0x74 |
| Row 1 + 11 x (N+1) | 0x1616 | 0x2c | 0x16 | 0x1676 | 0x2c | 0x76 |
| Row 1 + 12 x (N+1) | 0x1618 | 0x2c | 0x18 | 0x1678 | 0x2c | 0x78 |
| Row 1 + 13 x (N+1) | 0x161a | 0x2c | 0x1a | 0x167a | 0x2c | 0x7a |
| Row 1 + 14 x (N+1) | 0x161c | 0x2c | 0x1c | 0x167c | 0x2c | 0x7c |
| Row 1 + 15 x (N+1) | 0x161e | 0x2c | 0x1e | 0x167e | 0x2c | 0x7e |
| Row 1 + 16 x (N+1) | 0x1620 | 0x2c | 0x20 | 0x1680 | 0x2d | 0x00 |
| Row 1 + 17 x (N+1) | 0x1622 | 0x2c | 0x22 | 0x1682 | 0x2d | 0x02 |
| Row 1 + 18 x (N+1) | 0x1624 | 0x2c | 0x24 | 0x1684 | 0x2d | 0x04 |
| Row 1 + 19 x (N+1) | 0x1626 | 0x2c | 0x26 | 0x1686 | 0x2d | 0x06 |
| Row 1 + 20 x (N+1) | 0x1628 | 0x2c | 0x28 | 0x1688 | 0x2d | 0x08 |
| Row 1 + 21 x (N+1) | 0x162a | 0x2c | 0x2a | 0x168a | 0x2d | 0x0a |
| Row 1 + 22 x (N+1) | 0x162c | 0x2c | 0x2c | 0x168c | 0x2d | 0x0c |
| Row 1 + 23 x (N+1) | 0x162e | 0x2c | 0x2e | 0x168e | 0x2d | 0x0e |
| Row 1 + 24 x (N+1) | 0x1630 | 0x2c | 0x30 | 0x1690 | 0x2d | 0x10 |

**Flicker statistics: row sums, 48 rows/frame, N = flicker skip count (continued)**

| | Frame 1 | | | Frame 2 | | |
|---|---|---|---|---|---|---|
| **Description** | **Address** | **Block** | **Offset** | **Address** | **Block** | **Offset** |
| Row 1 + 25 x (N+1) | 0x1632 | 0x2c | 0x32 | 0x1692 | 0x2d | 0x12 |
| Row 1 + 26 x (N+1) | 0x1634 | 0x2c | 0x34 | 0x1694 | 0x2d | 0x14 |
| Row 1 + 27 x (N+1) | 0x1636 | 0x2c | 0x36 | 0x1696 | 0x2d | 0x16 |
| Row 1 + 28 x (N+1) | 0x1638 | 0x2c | 0x38 | 0x1698 | 0x2d | 0x18 |
| Row 1 + 29 x (N+1) | 0x163a | 0x2c | 0x3a | 0x169a | 0x2d | 0x1a |
| Row 1 + 30 x (N+1) | 0x163c | 0x2c | 0x3c | 0x169c | 0x2d | 0x1c |
| Row 1 + 31 x (N+1) | 0x163e | 0x2c | 0x3e | 0x169e | 0x2d | 0x1e |
| Row 1 + 32 x (N+1) | 0x1640 | 0x2c | 0x40 | 0x16a0 | 0x2d | 0x20 |
| Row 1 + 33 x (N+1) | 0x1642 | 0x2c | 0x42 | 0x16a2 | 0x2d | 0x22 |
| Row 1 + 34 x (N+1) | 0x1644 | 0x2c | 0x44 | 0x16a4 | 0x2d | 0x24 |
| Row 1 + 35 x (N+1) | 0x1646 | 0x2c | 0x46 | 0x16a6 | 0x2d | 0x26 |
| Row 1 + 36 x (N+1) | 0x1648 | 0x2c | 0x48 | 0x16a8 | 0x2d | 0x28 |
| Row 1 + 37 x (N+1) | 0x164a | 0x2c | 0x4a | 0x16aa | 0x2d | 0x2a |
| Row 1 + 38 x (N+1) | 0x164c | 0x2c | 0x4c | 0x16ac | 0x2d | 0x2c |
| Row 1 + 39 x (N+1) | 0x164e | 0x2c | 0x4e | 0x16ae | 0x2d | 0x2e |
| Row 1 + 40 x (N+1) | 0x1650 | 0x2c | 0x50 | 0x16b0 | 0x2d | 0x30 |
| Row 1 + 41 x (N+1) | 0x1652 | 0x2c | 0x52 | 0x16b2 | 0x2d | 0x32 |
| Row 1 + 42 x (N+1) | 0x1654 | 0x2c | 0x54 | 0x16b4 | 0x2d | 0x34 |
| Row 1 + 43 x (N+1) | 0x1656 | 0x2c | 0x56 | 0x16b6 | 0x2d | 0x36 |
| Row 1 + 44 x (N+1) | 0x1658 | 0x2c | 0x58 | 0x16b8 | 0x2d | 0x38 |
| Row 1 + 45 x (N+1) | 0x165a | 0x2c | 0x5a | 0x16ba | 0x2d | 0x3a |
| Row 1 + 46 x (N+1) | 0x165c | 0x2c | 0x5c | 0x16bc | 0x2d | 0x3c |
| Row 1 + 47 x (N+1) | 0x165e | 0x2c | 0x5e | 0x16be | 0x2d | 0x3e |

## Example 2

Flicker statistics, CIF window (296 rows), flicker skip count = 9, resulting in 30 row sums per frame.

**Flicker statistics: row sums, 30 rows/frame**

| | Frame 1 | | | Frame 2 | | |
|---|---|---|---|---|---|---|
| **Description** | **Address** | **Block** | **Offset** | **Address** | **Block** | **Offset** |
| Row 1 | 0x1600 | 0x2c | 0x00 | 0x163c | 0x2c | 0x3c |
| Row 11 | 0x1602 | 0x2c | 0x02 | 0x163e | 0x2c | 0x3e |
| Row 21 | 0x1604 | 0x2c | 0x04 | 0x1640 | 0x2c | 0x40 |
| Row 31 | 0x1606 | 0x2c | 0x06 | 0x1642 | 0x2c | 0x42 |
| Row 41 | 0x1608 | 0x2c | 0x08 | 0x1644 | 0x2c | 0x44 |
| Row 51 | 0x160a | 0x2c | 0x0a | 0x1646 | 0x2c | 0x46 |
| Row 61 | 0x160c | 0x2c | 0x0c | 0x1648 | 0x2c | 0x48 |
| Row 71 | 0x160e | 0x2c | 0x0e | 0x164a | 0x2c | 0x4a |
| Row 81 | 0x1610 | 0x2c | 0x10 | 0x164c | 0x2c | 0x4c |
| Row 91 | 0x1612 | 0x2c | 0x12 | 0x164e | 0x2c | 0x4e |
| Row 101 | 0x1614 | 0x2c | 0x14 | 0x1650 | 0x2c | 0x50 |
| Row 111 | 0x1616 | 0x2c | 0x16 | 0x1652 | 0x2c | 0x52 |
| Row 121 | 0x1618 | 0x2c | 0x18 | 0x1654 | 0x2c | 0x54 |
| Row 131 | 0x161a | 0x2c | 0x1a | 0x1656 | 0x2c | 0x56 |
| Row 141 | 0x161c | 0x2c | 0x1c | 0x1658 | 0x2c | 0x58 |
| Row 151 | 0x161e | 0x2c | 0x1e | 0x165a | 0x2c | 0x5a |
| Row 161 | 0x1620 | 0x2c | 0x20 | 0x165c | 0x2c | 0x5c |
| Row 171 | 0x1622 | 0x2c | 0x22 | 0x165e | 0x2c | 0x5e |
| Row 181 | 0x1624 | 0x2c | 0x24 | 0x1660 | 0x2c | 0x60 |
| Row 191 | 0x1626 | 0x2c | 0x26 | 0x1662 | 0x2c | 0x62 |
| Row 201 | 0x1628 | 0x2c | 0x28 | 0x1664 | 0x2c | 0x64 |
| Row 211 | 0x162a | 0x2c | 0x2a | 0x1666 | 0x2c | 0x66 |
| Row 221 | 0x162c | 0x2c | 0x2c | 0x1668 | 0x2c | 0x68 |
| Row 231 | 0x162e | 0x2c | 0x2e | 0x166a | 0x2c | 0x6a |
| Row 241 | 0x1630 | 0x2c | 0x30 | 0x166c | 0x2c | 0x6c |
| Row 251 | 0x1632 | 0x2c | 0x32 | 0x166e | 0x2c | 0x6e |
| Row 261 | 0x1634 | 0x2c | 0x34 | 0x1670 | 0x2c | 0x70 |
| Row 271 | 0x1636 | 0x2c | 0x36 | 0x1672 | 0x2c | 0x72 |
| Row 281 | 0x1638 | 0x2c | 0x38 | 0x1674 | 0x2c | 0x74 |
| Row 291 | 0x163a | 0x2c | 0x3a | 0x1676 | 0x2c | 0x76 |

# AV_RED

**Anti-vignetting, red pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| **Address** | **Mnemonic** | **Value** | **Block** | **Offset** |
| 0x1800 | AV_RED_0 | 0x0040 | 0x30 | 0x00 |
| 0x1802 | AV_RED_1 | 0x0040 | 0x30 | 0x02 |
| 0x1804 | AV_RED_2 | 0x0040 | 0x30 | 0x04 |
| 0x1806 | AV_RED_3 | 0x0040 | 0x30 | 0x06 |
| 0x1808 | AV_RED_4 | 0x0040 | 0x30 | 0x08 |
| 0x180a | AV_RED_5 | 0x0040 | 0x30 | 0x0a |
| 0x180c | AV_RED_6 | 0x0040 | 0x30 | 0x0c |
| 0x180e | AV_RED_7 | 0x0040 | 0x30 | 0x0e |
| 0x1810 | AV_RED_8 | 0x0040 | 0x30 | 0x10 |
| 0x1812 | AV_RED_9 | 0x0040 | 0x30 | 0x12 |
| 0x1814 | AV_RED_10 | 0x0040 | 0x30 | 0x14 |
| 0x1816 | AV_RED_11 | 0x0040 | 0x30 | 0x16 |
| 0x1818 | AV_RED_12 | 0x0040 | 0x30 | 0x18 |
| 0x181a | AV_RED_13 | 0x0040 | 0x30 | 0x1a |
| 0x181c | AV_RED_14 | 0x0040 | 0x30 | 0x1c |
| 0x181e | AV_RED_15 | 0x0040 | 0x30 | 0x1e |
| 0x1820 | AV_RED_16 | 0x0040 | 0x30 | 0x20 |
| 0x1822 | AV_RED_17 | 0x0040 | 0x30 | 0x22 |
| 0x1824 | AV_RED_18 | 0x0040 | 0x30 | 0x24 |
| 0x1826 | AV_RED_19 | 0x0040 | 0x30 | 0x26 |
| 0x1828 | AV_RED_20 | 0x0040 | 0x30 | 0x28 |
| 0x182a | AV_RED_21 | 0x0040 | 0x30 | 0x2a |
| 0x182c | AV_RED_22 | 0x0040 | 0x30 | 0x2c |
| 0x182e | AV_RED_23 | 0x0040 | 0x30 | 0x2e |
| 0x1830 | AV_RED_24 | 0x0040 | 0x30 | 0x30 |
| 0x1832 | AV_RED_25 | 0x0040 | 0x30 | 0x32 |
| 0x1834 | AV_RED_26 | 0x0040 | 0x30 | 0x34 |
| 0x1836 | AV_RED_27 | 0x0040 | 0x30 | 0x36 |
| 0x1838 | AV_RED_28 | 0x0040 | 0x30 | 0x38 |
| 0x183a | AV_RED_29 | 0x0040 | 0x30 | 0x3a |
| 0x183c | AV_RED_30 | 0x0040 | 0x30 | 0x3c |
| 0x183e | AV_RED_31 | 0x0040 | 0x30 | 0x3e |

# AV_GREEN

**Anti-vignetting, green pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| **Address** | **Mnemonic** | **Value** | **Block** | **Offset** |
| 0x1840 | AV_GREEN_0 | 0x0040 | 0x30 | 0x40 |
| 0x1842 | AV_GREEN_1 | 0x0040 | 0x30 | 0x42 |
| 0x1844 | AV_GREEN_2 | 0x0040 | 0x30 | 0x44 |
| 0x1846 | AV_GREEN_3 | 0x0040 | 0x30 | 0x46 |
| 0x1848 | AV_GREEN_4 | 0x0040 | 0x30 | 0x48 |
| 0x184a | AV_GREEN_5 | 0x0040 | 0x30 | 0x4a |
| 0x184c | AV_GREEN_6 | 0x0040 | 0x30 | 0x4c |
| 0x184e | AV_GREEN_7 | 0x0040 | 0x30 | 0x4e |
| 0x1850 | AV_GREEN_8 | 0x0040 | 0x30 | 0x50 |
| 0x1852 | AV_GREEN_9 | 0x0040 | 0x30 | 0x52 |
| 0x1854 | AV_GREEN_10 | 0x0040 | 0x30 | 0x54 |
| 0x1856 | AV_GREEN_11 | 0x0040 | 0x30 | 0x56 |
| 0x1858 | AV_GREEN_12 | 0x0040 | 0x30 | 0x58 |
| 0x185a | AV_GREEN_13 | 0x0040 | 0x30 | 0x5a |
| 0x185c | AV_GREEN_14 | 0x0040 | 0x30 | 0x5c |
| 0x185e | AV_GREEN_15 | 0x0040 | 0x30 | 0x5e |
| 0x1860 | AV_GREEN_16 | 0x0040 | 0x30 | 0x60 |
| 0x1862 | AV_GREEN_17 | 0x0040 | 0x30 | 0x62 |
| 0x1864 | AV_GREEN_18 | 0x0040 | 0x30 | 0x64 |
| 0x1866 | AV_GREEN_19 | 0x0040 | 0x30 | 0x66 |
| 0x1868 | AV_GREEN_20 | 0x0040 | 0x30 | 0x68 |
| 0x186a | AV_GREEN_21 | 0x0040 | 0x30 | 0x6a |
| 0x186c | AV_GREEN_22 | 0x0040 | 0x30 | 0x6c |
| 0x186e | AV_GREEN_23 | 0x0040 | 0x30 | 0x6e |
| 0x1870 | AV_GREEN_24 | 0x0040 | 0x30 | 0x70 |
| 0x1872 | AV_GREEN_25 | 0x0040 | 0x30 | 0x72 |
| 0x1874 | AV_GREEN_26 | 0x0040 | 0x30 | 0x74 |
| 0x1876 | AV_GREEN_27 | 0x0040 | 0x30 | 0x76 |
| 0x1878 | AV_GREEN_28 | 0x0040 | 0x30 | 0x78 |
| 0x187a | AV_GREEN_29 | 0x0040 | 0x30 | 0x7a |
| 0x187c | AV_GREEN_30 | 0x0040 | 0x30 | 0x7c |
| 0x187e | AV_GREEN_31 | 0x0040 | 0x30 | 0x7e |

# AV_BLUE

**Anti-vignetting, blue pixels**

| Full | | Default | Address | |
|---|---|---|---|---|
| **Address** | **Mnemonic** | **Value** | **Block** | **Offset** |
| 0x1880 | AV_BLUE_0 | 0x0040 | 0x31 | 0x00 |
| 0x1882 | AV_BLUE_1 | 0x0040 | 0x31 | 0x02 |
| 0x1884 | AV_BLUE_2 | 0x0040 | 0x31 | 0x04 |
| 0x1886 | AV_BLUE_3 | 0x0040 | 0x31 | 0x06 |
| 0x1888 | AV_BLUE_4 | 0x0040 | 0x31 | 0x08 |
| 0x188a | AV_BLUE_5 | 0x0040 | 0x31 | 0x0a |
| 0x188c | AV_BLUE_6 | 0x0040 | 0x31 | 0x0c |
| 0x188e | AV_BLUE_7 | 0x0040 | 0x31 | 0x0e |
| 0x1890 | AV_BLUE_8 | 0x0040 | 0x31 | 0x10 |
| 0x1892 | AV_BLUE_9 | 0x0040 | 0x31 | 0x12 |
| 0x1894 | AV_BLUE_10 | 0x0040 | 0x31 | 0x14 |
| 0x1896 | AV_BLUE_11 | 0x0040 | 0x31 | 0x16 |
| 0x1898 | AV_BLUE_12 | 0x0040 | 0x31 | 0x18 |
| 0x189a | AV_BLUE_13 | 0x0040 | 0x31 | 0x1a |
| 0x189c | AV_BLUE_14 | 0x0040 | 0x31 | 0x1c |
| 0x189e | AV_BLUE_15 | 0x0040 | 0x31 | 0x1e |
| 0x18a0 | AV_BLUE_16 | 0x0040 | 0x31 | 0x20 |
| 0x18a2 | AV_BLUE_17 | 0x0040 | 0x31 | 0x22 |
| 0x18a4 | AV_BLUE_18 | 0x0040 | 0x31 | 0x24 |
| 0x18a6 | AV_BLUE_19 | 0x0040 | 0x31 | 0x26 |
| 0x18a8 | AV_BLUE_20 | 0x0040 | 0x31 | 0x28 |
| 0x18aa | AV_BLUE_21 | 0x0040 | 0x31 | 0x2a |
| 0x18ac | AV_BLUE_22 | 0x0040 | 0x31 | 0x2c |
| 0x18ae | AV_BLUE_23 | 0x0040 | 0x31 | 0x2e |
| 0x18b0 | AV_BLUE_24 | 0x0040 | 0x31 | 0x30 |
| 0x18b2 | AV_BLUE_25 | 0x0040 | 0x31 | 0x32 |
| 0x18b4 | AV_BLUE_26 | 0x0040 | 0x31 | 0x34 |
| 0x18b6 | AV_BLUE_27 | 0x0040 | 0x31 | 0x36 |
| 0x18b8 | AV_BLUE_28 | 0x0040 | 0x31 | 0x38 |
| 0x18ba | AV_BLUE_29 | 0x0040 | 0x31 | 0x3a |
| 0x18bc | AV_BLUE_30 | 0x0040 | 0x31 | 0x3c |
| 0x18be | AV_BLUE_31 | 0x0040 | 0x31 | 0x3e |

# D
# Tonemaps (Gamma)



The ADCM-2700 can use tonemaps that are bottom-weighted or linear. The bottom-weighted tonemaps bring out more detail in the shadows. The tonemap function takes the 8-bit input and outputs a 10-bit number. The image pipeline will convert this back to eight bits later in the pipeline.

## Gamma Function

To calculate the value for the tonemap tables, use the following equation:

$$\text{Table Entry} = \frac{1024 \times \text{Input}^{\frac{1}{\gamma}}}{512^{\frac{1}{\gamma}}}$$

Where $\gamma$ is the gamma value, input is either the linear of bottom-weighted input value and table entry is the value to write into the tonemap table.

For example, if the gamma value is 2.2 and the input value is 112 (a value that is in both the linear and bottom-weighted inputs, the table value would be:

$$\text{Table Entry} = \frac{1024 \times 112^{\frac{1}{2.2}}}{512^{\frac{1}{2.2}}}$$

$$\text{Table Entry} = \frac{8745.06}{17.04}$$

$$\text{Table Entry} = 513.18 = 0\text{x}0201 \text{ hex}$$

## Linear and Bottom-Weighted Inputs

**Linear inputs**

| Index | Decimal | Hex | Index | Decimal | Hex | Index | Decimal | Hex |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0x0000 | 11 | 176 | 0x00b0 | 22 | 352 | 0x0160 |
| 1 | 16 | 0x0010 | 12 | 192 | 0x00c0 | 23 | 368 | 0x0170 |
| 2 | 32 | 0x0020 | 13 | 208 | 0x00d0 | 24 | 384 | 0x0180 |
| 3 | 48 | 0x0030 | 14 | 224 | 0x00e0 | 25 | 400 | 0x0190 |
| 4 | 64 | 0x0040 | 15 | 240 | 0x00f0 | 26 | 416 | 0x01a0 |
| 5 | 80 | 0x0050 | 16 | 256 | 0x0100 | 27 | 432 | 0x01b0 |
| 6 | 96 | 0x0060 | 17 | 272 | 0x0110 | 28 | 448 | 0x01c0 |
| 7 | 112 | 0x0070 | 18 | 288 | 0x0120 | 29 | 464 | 0x01d0 |
| 8 | 128 | 0x0080 | 19 | 304 | 0x0130 | 30 | 480 | 0x01e0 |
| 9 | 144 | 0x0090 | 20 | 320 | 0x0140 | 31 | 496 | 0x01f0 |
| 10 | 160 | 0x00a0 | 21 | 336 | 0x0150 | 32 | 512 | 0x0200 |

**Bottom-weighted inputs**

| Index | Decimal | Hex | Index | Decimal | Hex | Index | Decimal | Hex |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0x0000 | 11 | 14 | 0x000e | 22 | 96 | 0x0060 |
| 1 | 1 | 0x0001 | 12 | 16 | 0x0010 | 23 | 112 | 0x0070 |
| 2 | 2 | 0x0002 | 13 | 20 | 0x0014 | 24 | 128 | 0x0080 |
| 3 | 3 | 0x0003 | 14 | 24 | 0x0018 | 25 | 160 | 0x00a0 |
| 4 | 4 | 0x0004 | 15 | 28 | 0x001c | 26 | 192 | 0x00c0 |
| 5 | 5 | 0x0005 | 16 | 32 | 0x0020 | 27 | 224 | 0x00e0 |
| 6 | 6 | 0x0006 | 17 | 40 | 0x0028 | 28 | 256 | 0x0100 |
| 7 | 7 | 0x0007 | 18 | 48 | 0x0030 | 29 | 320 | 0x0140 |
| 8 | 8 | 0x0008 | 19 | 56 | 0x0038 | 30 | 384 | 0x0180 |
| 9 | 10 | 0x000a | 20 | 64 | 0x0040 | 31 | 448 | 0x01c0 |
| 10 | 12 | 0x000c | 21 | 80 | 0x0050 | 32 | 512 | 0x0200 |

## Address Offsets

The following are the address offsets for each of the tonemaps. The video and still tonemap “ALL” registers are written out to the tonemap RAM locations when changing modes.

**Table index number**

| | Color | | | |
|---|---|---|---|---|
| | **All** | **Red** | **Green** | **Blue** |
| Tonemap RAM | 0x1400 | 0x1480 | 0x1500 | 0x1580 |
| Still mode registers | 0x0202 | – | – | – |
| Video mode registers | 0x01c0 | – | – | – |

The following is the address offset table. Use the listed address offset for each table index number with the address offset for each color/mode combination to generate the RAM address for each tonemap table entry.

**Address offsets**

| Index | Offset | Index | Offset | Index | Offset |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x0016 | 22 | 0x002c |
| 1 | 0x0002 | 12 | 0x0018 | 23 | 0x002e |
| 2 | 0x0004 | 13 | 0x001a | 24 | 0x0030 |
| 3 | 0x0006 | 14 | 0x001c | 25 | 0x0032 |
| 4 | 0x0008 | 15 | 0x001e | 26 | 0x0034 |
| 5 | 0x000a | 16 | 0x0020 | 27 | 0x0036 |
| 6 | 0x000c | 17 | 0x0022 | 28 | 0x0038 |
| 7 | 0x000e | 18 | 0x0024 | 29 | 0x003a |
| 8 | 0x0010 | 19 | 0x0026 | 30 | 0x003c |
| 9 | 0x0012 | 20 | 0x0028 | 31 | 0x003e |
| 10 | 0x0014 | 21 | 0x002a | 32 | 0x0040 |

## Precalculated Tonemaps

The following tonemaps show the hexidecimal values for bottom-weighted tonemaps for various gamma values.

### Linear Gamma

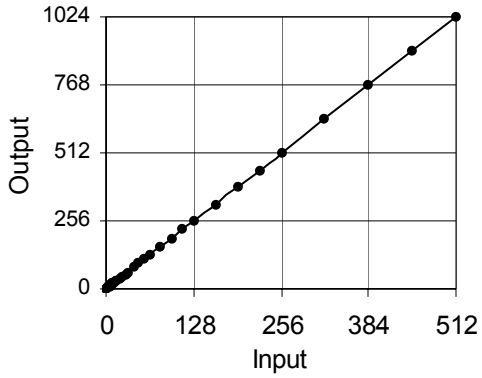



**Figure 99** Linear gamma curve

**Linear gamma table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x001c | 22 | 0x00c0 |
| 1 | 0x0002 | 12 | 0x0020 | 23 | 0x00e0 |
| 2 | 0x0004 | 13 | 0x0028 | 24 | 0x0100 |
| 3 | 0x0006 | 14 | 0x0030 | 25 | 0x0140 |
| 4 | 0x0008 | 15 | 0x0038 | 26 | 0x0180 |
| 5 | 0x000a | 16 | 0x0040 | 27 | 0x01c0 |
| 6 | 0x000c | 17 | 0x0050 | 28 | 0x0200 |
| 7 | 0x000e | 18 | 0x0060 | 29 | 0x0280 |
| 8 | 0x0010 | 19 | 0x0070 | 30 | 0x0300 |
| 9 | 0x0014 | 20 | 0x0080 | 31 | 0x0380 |
| 10 | 0x0018 | 21 | 0x00a0 | 32 | 0x0400 |

### sRGB Gamma, Default value

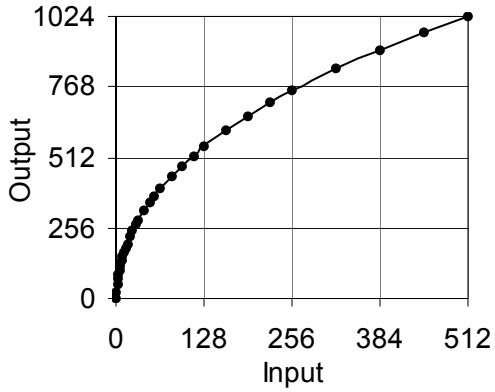



**Figure 100** sRBG gamma curve

**sRGB gamma table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00b8 | 22 | 0x01e1 |
| 1 | 0x0017 | 12 | 0x00c6 | 23 | 0x0205 |
| 2 | 0x0032 | 13 | 0x00df | 24 | 0x0225 |
| 3 | 0x0046 | 14 | 0x00f5 | 25 | 0x0261 |
| 4 | 0x0056 | 15 | 0x0109 | 26 | 0x0295 |
| 5 | 0x0064 | 16 | 0x011b | 27 | 0x02c5 |
| 6 | 0x0071 | 17 | 0x013d | 28 | 0x02f1 |
| 7 | 0x007c | 18 | 0x015a | 29 | 0x033f |
| 8 | 0x0086 | 19 | 0x0175 | 30 | 0x0385 |
| 9 | 0x0099 | 20 | 0x018d | 31 | 0x03c5 |
| 10 | 0x00a9 | 21 | 0x01ba | 32 | 0x0400 |

## Gamma 1.4

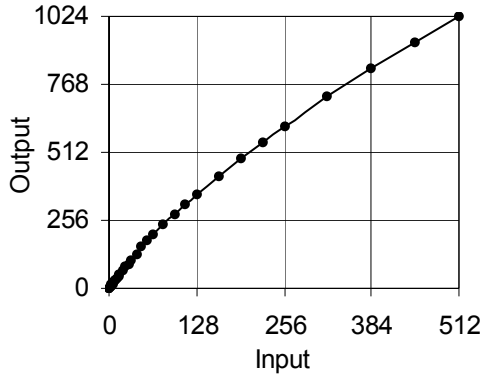



**Figure 101** Gamma 1.4 curve

**Gamma 1.4 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x004e | 22 | 0x0135 |
| 1 | 0x000b | 12 | 0x0056 | 23 | 0x0159 |
| 2 | 0x0013 | 13 | 0x0065 | 24 | 0x017c |
| 3 | 0x001a | 14 | 0x0073 | 25 | 0x01be |
| 4 | 0x0020 | 15 | 0x0080 | 26 | 0x01fc |
| 5 | 0x0025 | 16 | 0x008d | 27 | 0x0237 |
| 6 | 0x002a | 17 | 0x00a5 | 28 | 0x0270 |
| 7 | 0x002f | 18 | 0x00bc | 29 | 0x02db |
| 8 | 0x0034 | 19 | 0x00d2 | 30 | 0x0341 |
| 9 | 0x003d | 20 | 0x00e7 | 31 | 0x03a2 |
| 10 | 0x0046 | 21 | 0x010f | 32 | 0x0400 |

## Gamma 1.5

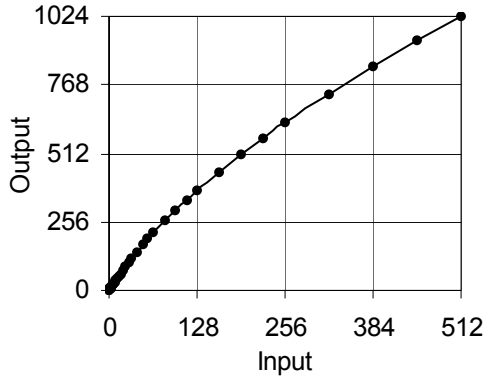



**Figure 102** Gamma 1.5 curve

**Gamma 1.5 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x005c | 22 | 0x014f |
| 1 | 0x0010 | 12 | 0x0065 | 23 | 0x0173 |
| 2 | 0x0019 | 13 | 0x0075 | 24 | 0x0196 |
| 3 | 0x0021 | 14 | 0x0085 | 25 | 0x01d7 |
| 4 | 0x0028 | 15 | 0x0093 | 26 | 0x0214 |
| 5 | 0x002e | 16 | 0x00a1 | 27 | 0x024e |
| 6 | 0x0034 | 17 | 0x00bb | 28 | 0x0285 |
| 7 | 0x003a | 18 | 0x00d3 | 29 | 0x02ec |
| 8 | 0x0040 | 19 | 0x00ea | 30 | 0x034d |
| 9 | 0x004a | 20 | 0x0100 | 31 | 0x03a8 |
| 10 | 0x0053 | 21 | 0x0129 | 32 | 0x0400 |

## Gamma 1.6

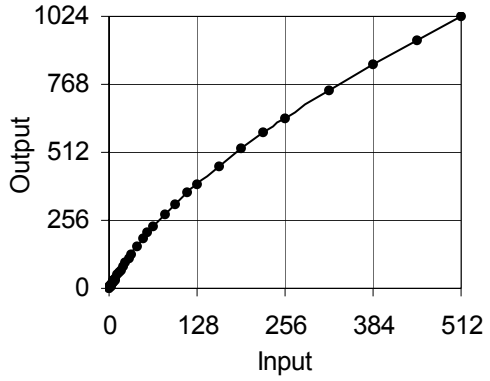



**Figure 103** Gamma 1.6 curve

**Gamma 1.6 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x006b | 22 | 0x0167 |
| 1 | 0x0014 | 12 | 0x0075 | 23 | 0x018c |
| 2 | 0x0020 | 13 | 0x0086 | 24 | 0x01ae |
| 3 | 0x0029 | 14 | 0x0097 | 25 | 0x01ee |
| 4 | 0x0031 | 15 | 0x00a6 | 26 | 0x022a |
| 5 | 0x0038 | 16 | 0x00b5 | 27 | 0x0262 |
| 6 | 0x003f | 17 | 0x00d0 | 28 | 0x0297 |
| 7 | 0x0046 | 18 | 0x00e9 | 29 | 0x02fb |
| 8 | 0x004c | 19 | 0x0100 | 30 | 0x0357 |
| 9 | 0x0057 | 20 | 0x0117 | 31 | 0x03ae |
| 10 | 0x0062 | 21 | 0x0140 | 32 | 0x0400 |

## Gamma 1.7

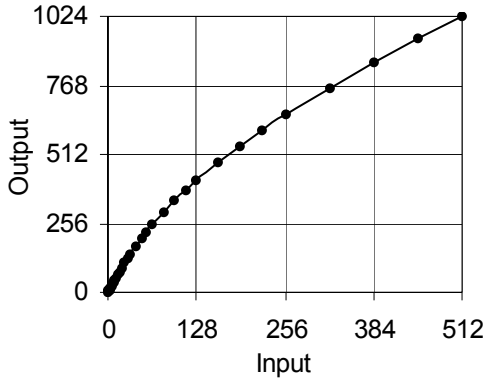



**Figure 104** Gamma 1.7 curve

**Gamma 1.7 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x007b | 22 | 0x017e |
| 1 | 0x001a | 12 | 0x0085 | 23 | 0x01a2 |
| 2 | 0x0027 | 13 | 0x0098 | 24 | 0x01c5 |
| 3 | 0x0031 | 14 | 0x00a9 | 25 | 0x0204 |
| 4 | 0x003a | 15 | 0x00b9 | 26 | 0x023f |
| 5 | 0x0043 | 16 | 0x00c8 | 27 | 0x0275 |
| 6 | 0x004a | 17 | 0x00e4 | 28 | 0x02a9 |
| 7 | 0x0051 | 18 | 0x00fe | 29 | 0x0308 |
| 8 | 0x0058 | 19 | 0x0116 | 30 | 0x0360 |
| 9 | 0x0065 | 20 | 0x012d | 31 | 0x03b2 |
| 10 | 0x0070 | 21 | 0x0157 | 32 | 0x0400 |

## Gamma 1.8

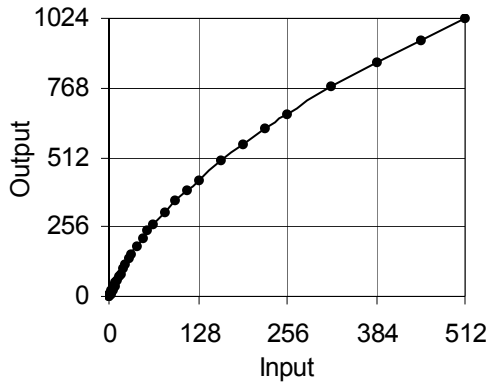



**Figure 105** Gamma 1.8 curve

**Gamma 1.8 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x008a | 22 | 0x0194 |
| 1 | 0x0020 | 12 | 0x0095 | 23 | 0x01b8 |
| 2 | 0x002f | 13 | 0x00a9 | 24 | 0x01da |
| 3 | 0x003a | 14 | 0x00bb | 25 | 0x0218 |
| 4 | 0x0045 | 15 | 0x00cb | 26 | 0x0251 |
| 5 | 0x004e | 16 | 0x00db | 27 | 0x0286 |
| 6 | 0x0056 | 17 | 0x00f8 | 28 | 0x02b8 |
| 7 | 0x005e | 18 | 0x0112 | 29 | 0x0314 |
| 8 | 0x0065 | 19 | 0x012b | 30 | 0x0368 |
| 9 | 0x0073 | 20 | 0x0142 | 31 | 0x03b6 |
| 10 | 0x007f | 21 | 0x016d | 32 | 0x0400 |

## Gamma 1.9

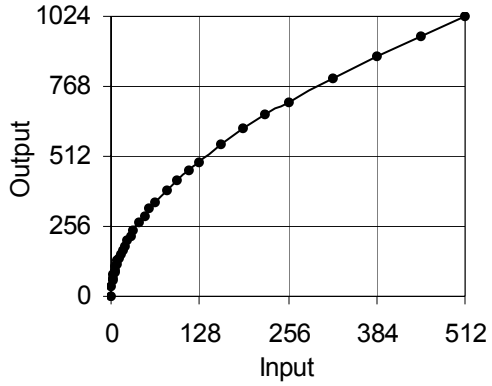



**Figure 106** Gamma 1.9 curve

**Gamma 1.9 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x009a | 22 | 0x01a8 |
| 1 | 0x0026 | 12 | 0x00a5 | 23 | 0x01cc |
| 2 | 0x0037 | 13 | 0x00b9 | 24 | 0x01ed |
| 3 | 0x0044 | 14 | 0x00cc | 25 | 0x022b |
| 4 | 0x004f | 15 | 0x00dd | 26 | 0x0263 |
| 5 | 0x0059 | 16 | 0x00ed | 27 | 0x0296 |
| 6 | 0x0062 | 17 | 0x010b | 28 | 0x02c6 |
| 7 | 0x006a | 18 | 0x0126 | 29 | 0x031f |
| 8 | 0x0072 | 19 | 0x013f | 30 | 0x0370 |
| 9 | 0x0081 | 20 | 0x0156 | 31 | 0x03ba |
| 10 | 0x008e | 21 | 0x0181 | 32 | 0x0400 |

## Gamma 2.0

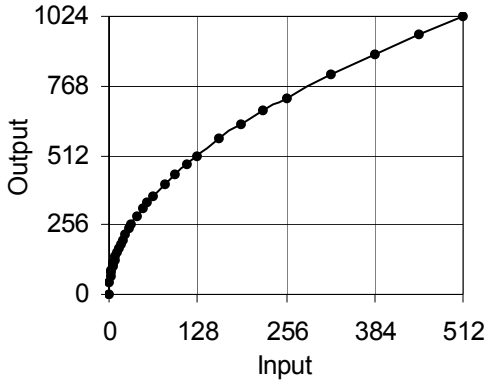



**Figure 107** Gamma 2.0 curve

**Gamma 2.0 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00a9 | 22 | 0x01bb |
| 1 | 0x002d | 12 | 0x00b5 | 23 | 0x01de |
| 2 | 0x0040 | 13 | 0x00ca | 24 | 0x0200 |
| 3 | 0x004e | 14 | 0x00dd | 25 | 0x023c |
| 4 | 0x005a | 15 | 0x00ef | 26 | 0x0273 |
| 5 | 0x0065 | 16 | 0x0100 | 27 | 0x02a5 |
| 6 | 0x006e | 17 | 0x011e | 28 | 0x02d4 |
| 7 | 0x0077 | 18 | 0x0139 | 29 | 0x0329 |
| 8 | 0x0080 | 19 | 0x0152 | 30 | 0x0376 |
| 9 | 0x008f | 20 | 0x016a | 31 | 0x03bd |
| 10 | 0x009c | 21 | 0x0194 | 32 | 0x0400 |

## Gamma 2.1

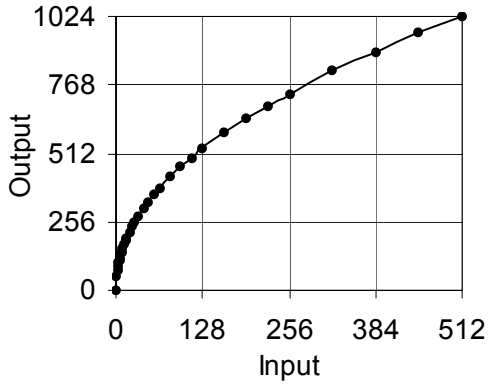



**Figure 108** Gamma 2.1 curve

**Gamma 2.1 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00b8 | 22 | 0x01cd |
| 1 | 0x0034 | 12 | 0x00c4 | 23 | 0x01f0 |
| 2 | 0x0049 | 13 | 0x00da | 24 | 0x0211 |
| 3 | 0x0058 | 14 | 0x00ee | 25 | 0x024c |
| 4 | 0x0065 | 15 | 0x0100 | 26 | 0x0281 |
| 5 | 0x0070 | 16 | 0x0111 | 27 | 0x02b2 |
| 6 | 0x007b | 17 | 0x0130 | 28 | 0x02e0 |
| 7 | 0x0084 | 18 | 0x014b | 29 | 0x0332 |
| 8 | 0x008d | 19 | 0x0164 | 30 | 0x037c |
| 9 | 0x009d | 20 | 0x017c | 31 | 0x03c0 |
| 10 | 0x00ab | 21 | 0x01a7 | 32 | 0x0400 |

## Gamma 2.2

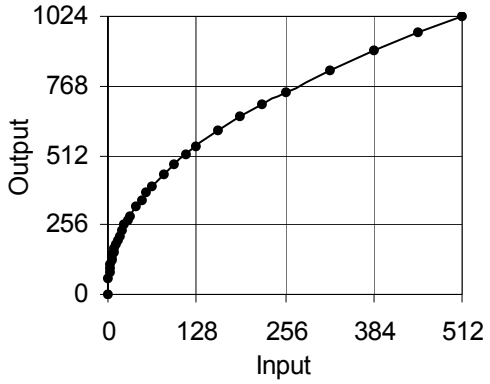



**Figure 109** Gamma 2.2 curve

**Gamma 2.2 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00c7 | 22 | 0x01de |
| 1 | 0x003c | 12 | 0x00d3 | 23 | 0x0201 |
| 2 | 0x0052 | 13 | 0x00ea | 24 | 0x0221 |
| 3 | 0x0063 | 14 | 0x00fe | 25 | 0x025b |
| 4 | 0x0070 | 15 | 0x0111 | 26 | 0x028f |
| 5 | 0x007c | 16 | 0x0122 | 27 | 0x02bf |
| 6 | 0x0087 | 17 | 0x0141 | 28 | 0x02eb |
| 7 | 0x0091 | 18 | 0x015d | 29 | 0x033b |
| 8 | 0x0094 | 19 | 0x0176 | 30 | 0x0382 |
| 9 | 0x00ab | 20 | 0x018d | 31 | 0x03c3 |
| 10 | 0x00b9 | 21 | 0x01b8 | 32 | 0x0400 |

## Gamma 2.3

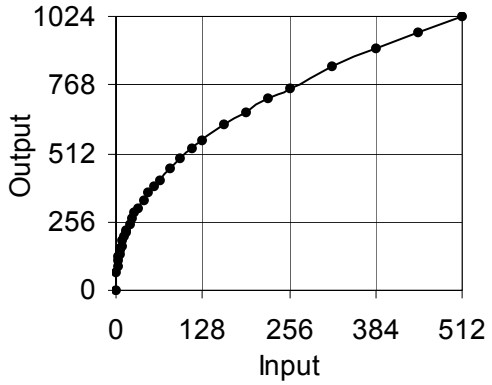



**Figure 110** Gamma 2.3 curve

**Gamma 2.3 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00d6 | 22 | 0x01ee |
| 1 | 0x0043 | 12 | 0x00e2 | 23 | 0x0210 |
| 2 | 0x005b | 13 | 0x00fa | 24 | 0x0230 |
| 3 | 0x006d | 14 | 0x010e | 25 | 0x0269 |
| 4 | 0x007c | 15 | 0x0121 | 26 | 0x029c |
| 5 | 0x0088 | 16 | 0x0132 | 27 | 0x02ca |
| 6 | 0x0094 | 17 | 0x0151 | 28 | 0x02f5 |
| 7 | 0x009e | 18 | 0x016d | 29 | 0x0342 |
| 8 | 0x00a7 | 19 | 0x0187 | 30 | 0x0387 |
| 9 | 0x00b8 | 20 | 0x019e | 31 | 0x03c6 |
| 10 | 0x00c8 | 21 | 0x01c8 | 32 | 0x0400 |

## Gamma 2.4

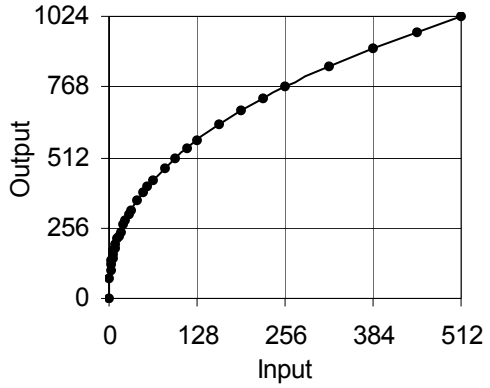



**Figure 111** Gamma 2.4 curve

**Gamma 2.4 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00e4 | 22 | 0x01fd |
| 1 | 0x004c | 12 | 0x00f1 | 23 | 0x021f |
| 2 | 0x0065 | 13 | 0x0109 | 24 | 0x023e |
| 3 | 0x0078 | 14 | 0x011e | 25 | 0x0276 |
| 4 | 0x0087 | 15 | 0x0131 | 26 | 0x02a8 |
| 5 | 0x0094 | 16 | 0x0142 | 27 | 0x02d5 |
| 6 | 0x00a0 | 17 | 0x0161 | 28 | 0x02ff |
| 7 | 0x00ab | 18 | 0x017d | 29 | 0x0349 |
| 8 | 0x00b5 | 19 | 0x0197 | 30 | 0x038c |
| 9 | 0x00c6 | 20 | 0x01ae | 31 | 0x03c8 |
| 10 | 0x00d6 | 21 | 0x01d8 | 32 | 0x0400 |

## Gamma 2.5

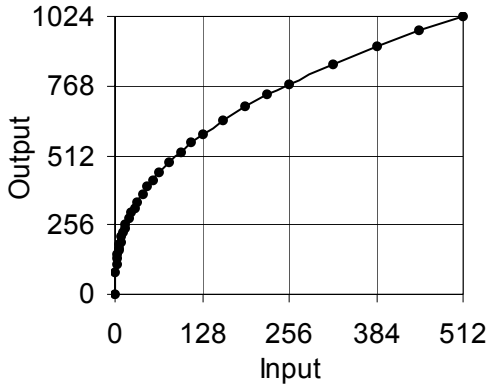



**Figure 112** Gamma 2.5 curve

**Gamma 2.5 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x00f2 | 22 | 0x020c |
| 1 | 0x0054 | 12 | 0x00ff | 23 | 0x022d |
| 2 | 0x006f | 13 | 0x0117 | 24 | 0x024c |
| 3 | 0x0083 | 14 | 0x012d | 25 | 0x0283 |
| 4 | 0x0093 | 15 | 0x0140 | 26 | 0x02b3 |
| 5 | 0x00a0 | 16 | 0x0151 | 27 | 0x02df |
| 6 | 0x00ac | 17 | 0x0171 | 28 | 0x0308 |
| 7 | 0x00b7 | 18 | 0x018d | 29 | 0x0350 |
| 8 | 0x00c2 | 19 | 0x01a6 | 30 | 0x0390 |
| 9 | 0x00d4 | 20 | 0x01bd | 31 | 0x03ca |
| 10 | 0x00e4 | 21 | 0x01e7 | 32 | 0x0400 |

## Gamma 2.6

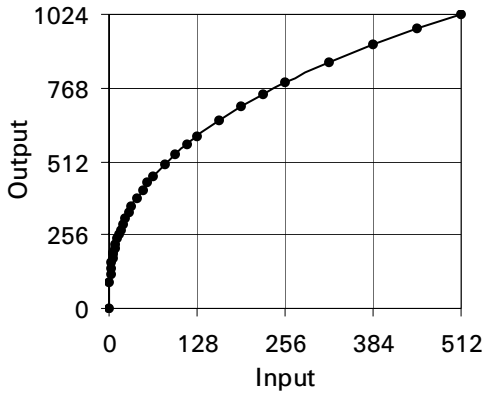



**Figure 113** Gamma 2.6 curve

**Gamma 2.6 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x0100 | 22 | 0x0219 |
| 1 | 0x005c | 12 | 0x010e | 23 | 0x023a |
| 2 | 0x0079 | 13 | 0x0126 | 24 | 0x0258 |
| 3 | 0x008d | 14 | 0x013b | 25 | 0x028e |
| 4 | 0x009e | 15 | 0x014e | 26 | 0x02be |
| 5 | 0x00ac | 16 | 0x0160 | 27 | 0x02e9 |
| 6 | 0x00b9 | 17 | 0x0180 | 28 | 0x0310 |
| 7 | 0x00c4 | 18 | 0x019c | 29 | 0x0356 |
| 8 | 0x00ce | 19 | 0x01b5 | 30 | 0x0394 |
| 9 | 0x00e1 | 20 | 0x01cc | 31 | 0x03cc |
| 10 | 0x00f1 | 21 | 0x01f5 | 32 | 0x0400 |

## Gamma 2.7

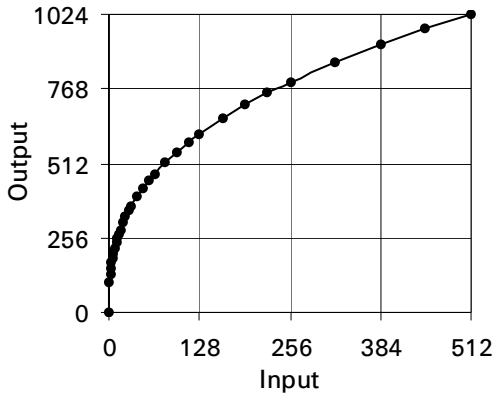



**Figure 114** Gamma 2.7 curve

**Gamma 2.7 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x010d | 22 | 0x0226 |
| 1 | 0x0065 | 12 | 0x011b | 23 | 0x0247 |
| 2 | 0x0083 | 13 | 0x0134 | 24 | 0x0264 |
| 3 | 0x0098 | 14 | 0x0149 | 25 | 0x0299 |
| 4 | 0x00a9 | 15 | 0x015d | 26 | 0x02c8 |
| 5 | 0x00b8 | 16 | 0x016e | 27 | 0x02f1 |
| 6 | 0x00c5 | 17 | 0x018e | 28 | 0x0318 |
| 7 | 0x00d0 | 18 | 0x01aa | 29 | 0x035c |
| 8 | 0x00db | 19 | 0x01c3 | 30 | 0x0398 |
| 9 | 0x00ee | 20 | 0x01da | 31 | 0x03ce |
| 10 | 0x00ff | 21 | 0x0202 | 32 | 0x0400 |

## Gamma 2.8

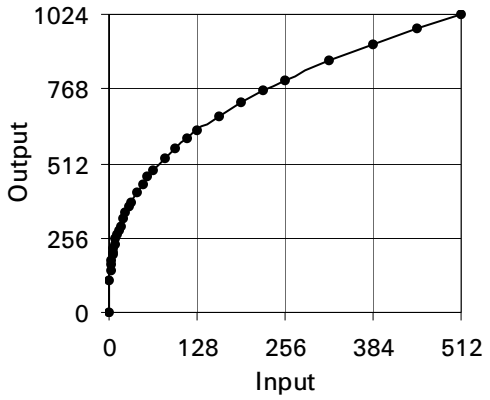



**Figure 115** Gamma 2.8 curve

**Gamma 2.8 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x011b | 22 | 0x0233 |
| 1 | 0x006e | 12 | 0x0128 | 23 | 0x0253 |
| 2 | 0x008d | 13 | 0x0141 | 24 | 0x0270 |
| 3 | 0x00a3 | 14 | 0x0157 | 25 | 0x02a3 |
| 4 | 0x00b5 | 15 | 0x016a | 26 | 0x02d1 |
| 5 | 0x00c4 | 16 | 0x017c | 27 | 0x02fa |
| 6 | 0x00d1 | 17 | 0x019b | 28 | 0x031f |
| 7 | 0x00dd | 18 | 0x01b7 | 29 | 0x0361 |
| 8 | 0x00e7 | 19 | 0x01d0 | 30 | 0x039c |
| 9 | 0x00fb | 20 | 0x01e7 | 31 | 0x03d0 |
| 10 | 0x010b | 21 | 0x020f | 32 | 0x0400 |

## Gamma 2.9

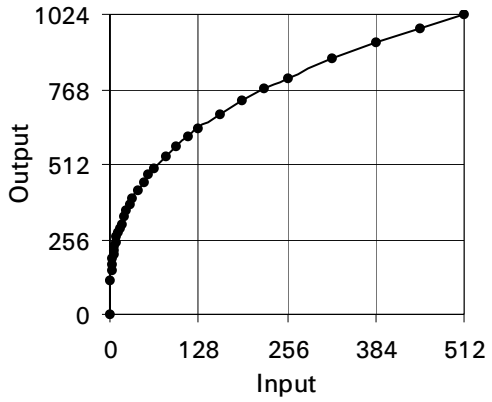



**Figure 116** Gamma 2.9 curve

**Gamma 2.9 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x0127 | 22 | 0x023e |
| 1 | 0x0077 | 12 | 0x0135 | 23 | 0x025e |
| 2 | 0x0097 | 13 | 0x014e | 24 | 0x027a |
| 3 | 0x00ae | 14 | 0x0164 | 25 | 0x02ad |
| 4 | 0x00c0 | 15 | 0x0177 | 26 | 0x02da |
| 5 | 0x00cf | 16 | 0x0189 | 27 | 0x0302 |
| 6 | 0x00dd | 17 | 0x01a9 | 28 | 0x0326 |
| 7 | 0x00e9 | 18 | 0x01c4 | 29 | 0x0366 |
| 8 | 0x00f4 | 19 | 0x01dd | 30 | 0x039f |
| 9 | 0x0107 | 20 | 0x01f3 | 31 | 0x03d1 |
| 10 | 0x0118 | 21 | 0x021b | 32 | 0x0400 |

## Gamma 3.0

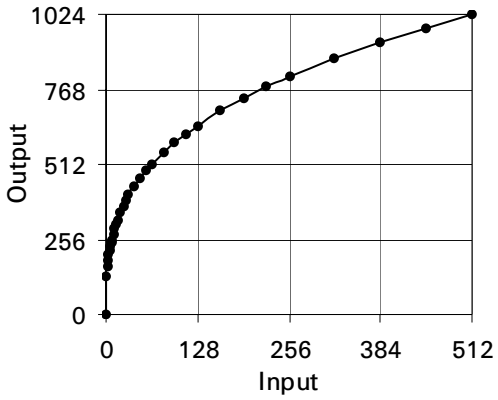



**Figure 117** Gamma 3.0 curve

**Gamma 3.0 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x0134 | 22 | 0x024a |
| 1 | 0x0080 | 12 | 0x0142 | 23 | 0x0268 |
| 2 | 0x00a1 | 13 | 0x015b | 24 | 0x0285 |
| 3 | 0x00b8 | 14 | 0x0171 | 25 | 0x02b6 |
| 4 | 0x00cb | 15 | 0x0184 | 26 | 0x02e2 |
| 5 | 0x00da | 16 | 0x0196 | 27 | 0x0309 |
| 6 | 0x00e8 | 17 | 0x01b5 | 28 | 0x032c |
| 7 | 0x00f4 | 18 | 0x01d1 | 29 | 0x036b |
| 8 | 0x0100 | 19 | 0x01e9 | 30 | 0x03a2 |
| 9 | 0x0113 | 20 | 0x0200 | 31 | 0x03d3 |
| 10 | 0x0125 | 21 | 0x0227 | 32 | 0x0400 |

## Gamma 3.1

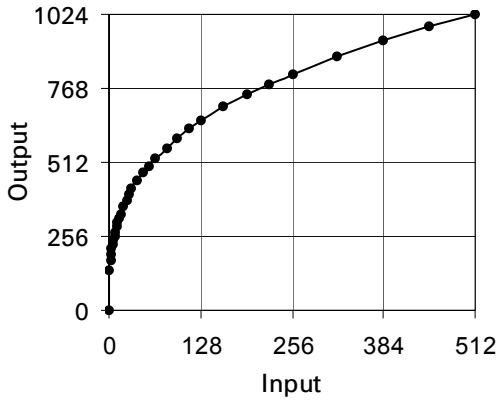



**Figure 118** Gamma 3.1 curve

**Gamma 3.1 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x0140 | 22 | 0x0254 |
| 1 | 0x0088 | 12 | 0x014e | 23 | 0x0273 |
| 2 | 0x00ab | 13 | 0x0167 | 24 | 0x028e |
| 3 | 0x00c3 | 14 | 0x017d | 25 | 0x02bf |
| 4 | 0x00d6 | 15 | 0x0191 | 26 | 0x02ea |
| 5 | 0x00e6 | 16 | 0x01a2 | 27 | 0x0310 |
| 6 | 0x00f3 | 17 | 0x01c1 | 28 | 0x0332 |
| 7 | 0x0100 | 18 | 0x01dd | 29 | 0x036f |
| 8 | 0x010b | 19 | 0x01f5 | 30 | 0x03a5 |
| 9 | 0x011f | 20 | 0x020b | 31 | 0x03d4 |
| 10 | 0x0131 | 21 | 0x0232 | 32 | 0x0400 |

## Gamma 3.2

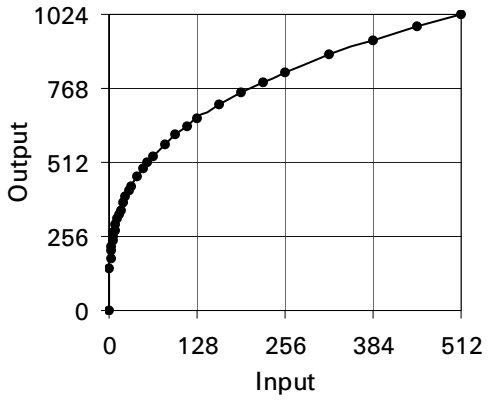



**Figure 119** Gamma 3.2 curve

**Gamma 3.2 table**

| Index | Value | Index | Value | Index | Value |
|---|---|---|---|---|---|
| 0 | 0x0000 | 11 | 0x014c | 22 | 0x025e |
| 1 | 0x0091 | 12 | 0x015a | 23 | 0x027c |
| 2 | 0x00b5 | 13 | 0x0173 | 24 | 0x0297 |
| 3 | 0x00cd | 14 | 0x0189 | 25 | 0x02c7 |
| 4 | 0x00e0 | 15 | 0x019c | 26 | 0x02f1 |
| 5 | 0x00f1 | 16 | 0x01ae | 27 | 0x0316 |
| 6 | 0x00ff | 17 | 0x01cd | 28 | 0x0338 |
| 7 | 0x010b | 18 | 0x01e8 | 29 | 0x0374 |
| 8 | 0x0117 | 19 | 0x0200 | 30 | 0x03a7 |
| 9 | 0x012b | 20 | 0x0216 | 31 | 0x03d6 |
| 10 | 0x013c | 21 | 0x023d | 32 | 0x0400 |

# E
# Image Zones

## Camera Image Zones

The camera module divides the sensor into a series of zones for calculating the peak pixel intensity and pixel intensity sum statistics. Each zone can be weighted by a factor of 0, ¼, ½, or 1.

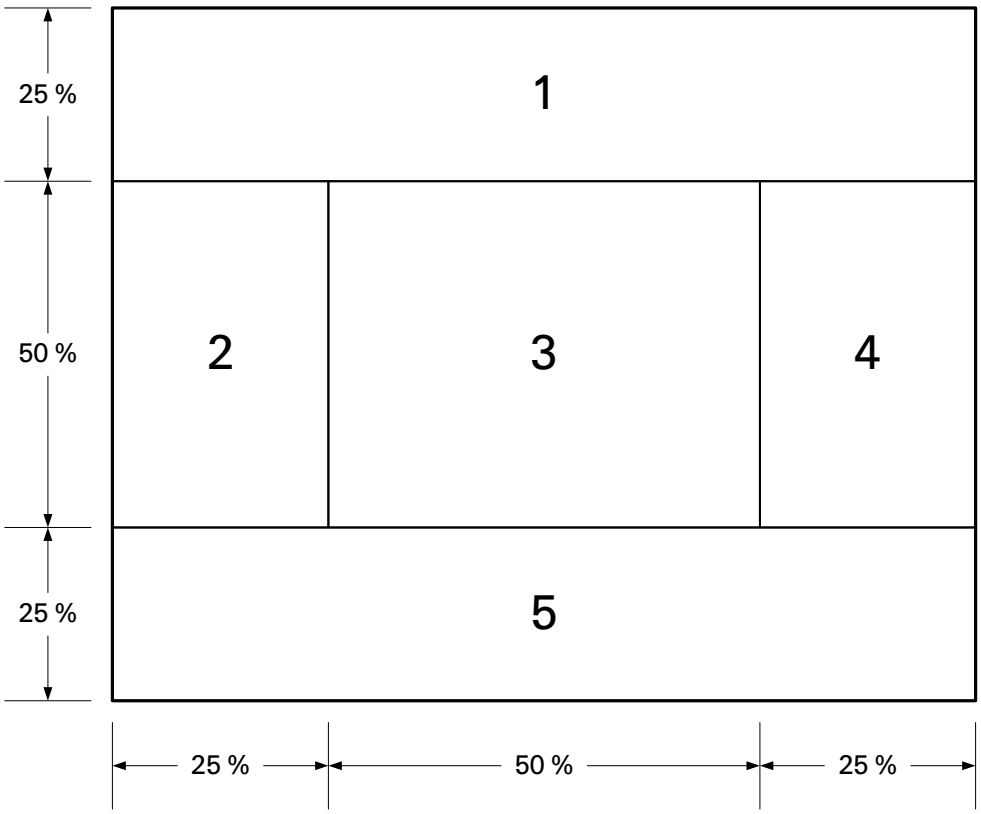



**Figure 120** Image zones

# Index

## Numerics



## A

## B

## C

## D



## E

## F

## G



## H

## I

## J



## L



## M

## N



## O

## P

## Q



## R

## S

## T

## U



## V

## W



## X



## Y



## Z