cc3 Reference Manual

1.0

Generated by Doxygen 1.5.1

Sat May 19 12:58:10 2007

Contents

1 cc3 Data Structure Index **1**

1.1 cc3 Data Structures 1

2 cc3 File Index **3**

2.1 cc3 File List 3

3 cc3 Data Structure Documentation **5**

3.1 cc3_frame_t Struct Reference 5

3.2 cc3_pixel_t Struct Reference 7

4 cc3 File Documentation **9**

4.1 include/cc3.h File Reference 9

Chapter 1

cc3 Data Structure Index

1.1 cc3 Data Structures

Here are the data structures with brief descriptions:

cc3_frame_t (Frame structure) 5
cc3_pixel_t (Simple 3-channel pixel structure) 7

Chapter 2

cc3 File Index

2.1 cc3 File List

Here is a list of all documented files with brief descriptions:

include/cc3.h (The core of the cc3 system) 9
lib/cc3-ilp/**cc3_color_info.h** . . . . . **??**
lib/cc3-ilp/**cc3_color_track.h** . . . . . **??**
lib/cc3-ilp/**cc3_connected_component.h** . . . . . **??**
lib/cc3-ilp/**cc3_conv.h** . . . . . **??**
lib/cc3-ilp/**cc3_frame_diff.h** . . . . . **??**
lib/cc3-ilp/**cc3_histogram.h** . . . . . **??**
lib/cc3-ilp/**cc3_hsv.h** . . . . . **??**
lib/cc3-ilp/**cc3_ilp.h** . . . . . **??**
lib/cc3-ilp/**cc3_img_writer.h** . . . . . **??**
lib/cc3-ilp/**cc3_jpg.h** . . . . . **??**
lib/cc3-ilp/**cc3_math.h** . . . . . **??**

Chapter 3

cc3 Data Structure Documentation

3.1 cc3_frame_t Struct Reference

3.1.1 Detailed Description

Frame structure.

See also:

cc3_g_pixbuf_frame for the main use of this structure.

Data Fields

- uint16_t raw_width
  Native width.
- uint16_t raw_height
  Native height.
- uint16_t x0
  Left horizontal clipping boundary.
- uint16_t y0
  Top vertical clipping boundary.
- uint16_t x1

Right horizontal clipping boundary.

- uint16_t y1

  Bottom horizontal clipping boundary.

- uint16_t y_loc

  Currently selected row.

- uint8_t x_step

  Horizontal subsampling value.

- uint8_t y_step

  Vertical subsampling value.

- cc3_channel_t coi

  Channel of interest.

- cc3_subsample_mode_t subsample_mode

  Subsampling mode.

- uint16_t width

  Width of clipping region.

- uint16_t height

  Height of clipping region.

- uint8_t channels

  Number of channels.

3.2 cc3_pixel_t Struct Reference

3.2.1 Detailed Description

Simple 3-channel pixel structure.

Data Fields

- uint8_t channel [3]

  Components of a single pixel.

Chapter 4

cc3 File Documentation

4.1 include/cc3.h File Reference

4.1.1 Detailed Description

The core of the cc3 system.

Data Structures

- struct cc3_frame_t

  Frame structure.

- struct cc3_pixel_t

  Simple 3-channel pixel structure.

Defines

- #define CC3_API_MAJOR_VERSION 1

  Major version number of API.

- #define CC3_API_MINOR_VERSION 0

  Minor version number of API.

Enumerations

- enum cc3_camera_resolution_t {
  CC3_CAMERA_RESOLUTION_LOW = 0,
  CC3_CAMERA_RESOLUTION_HIGH = 1 }
  Allowed resolutions for the camera device.

- enum cc3_channel_t {
  CC3_CHANNEL_SINGLE = 0,
  CC3_CHANNEL_RED = 0,
  CC3_CHANNEL_GREEN = 1,
  CC3_CHANNEL_BLUE = 2,
  CC3_CHANNEL_Y = 0,
  CC3_CHANNEL_CR = 1,
  CC3_CHANNEL_CB = 2,
  CC3_CHANNEL_HUE = 0,
  CC3_CHANNEL_SAT = 1,
  CC3_CHANNEL_VAL = 2,
  CC3_CHANNEL_ALL }
  Allowable channel values for channel of interest selection and other functions.

- enum cc3_colorspace_t {
  CC3_COLORSPACE_YCRCB = 0,
  CC3_COLORSPACE_RGB = 1 }
  Colorspace selector.

- enum cc3_subsample_mode_t {
  CC3_SUBSAMPLE_NEAREST,
  CC3_SUBSAMPLE_MEAN,
  CC3_SUBSAMPLE_RANDOM }
  Subsampling modes.

- enum cc3_uart_rate_t {
  CC3_UART_RATE_300 = 300,
  CC3_UART_RATE_600 = 600,
  CC3_UART_RATE_1200 = 1200,
  CC3_UART_RATE_2400 = 2400,

CC3_UART_RATE_4800 = 4800,
CC3_UART_RATE_9600 = 9600,
CC3_UART_RATE_14400 = 14400,
CC3_UART_RATE_19200 = 19200,
CC3_UART_RATE_38400 = 38400,
CC3_UART_RATE_57600 = 57600,
CC3_UART_RATE_115200 = 115200,
CC3_UART_RATE_230400 = 230400 }

UART speeds.

- enum cc3_uart_mode_t {
  CC3_UART_MODE_8N1,
  CC3_UART_MODE_7N1,
  CC3_UART_MODE_8N2,
  CC3_UART_MODE_7N2,
  CC3_UART_MODE_8E1,
  CC3_UART_MODE_7E1,
  CC3_UART_MODE_8E2,
  CC3_UART_MODE_7E2,
  CC3_UART_MODE_8O1,
  CC3_UART_MODE_7O1,
  CC3_UART_MODE_8O2,
  CC3_UART_MODE_7O2 }

  UART modes.

- enum cc3_uart_binmode_t {
  CC3_UART_BINMODE_BINARY,
  CC3_UART_BINMODE_TEXT }

  UART binary/text mode selection values.

- enum cc3_gpio_mode_t {
  CC3_GPIO_MODE_INPUT,
  CC3_GPIO_MODE_OUTPUT,
  CC3_GPIO_MODE_SERVO }

  GPIO configuration values.

Functions

- bool cc3_camera_init (void)

  Initialize camera hardware.

- void cc3_filesystem_init (void)

  Initialize the filesystem drivers.

- void cc3_pixbuf_load (void)

  Take a picture with the camera and load it into the internal pixbuf.

- uint8_t * cc3_malloc_rows (uint32_t rows)

  Use malloc() to allocate a number of rows of the correct size based on the values in cc3_g_pixbuf_frame.

- int cc3_pixbuf_read_rows (void *mem, uint32_t rows)

  Do a row-by-row copy from the pixbuf into a block of memory.

- void cc3_pixbuf_rewind (void)

  Rewind the pixbuf to the beginning without changing the image data stored in it.

- bool cc3_pixbuf_frame_set_roi (int16_t x_0, int16_t y_0, int16_t x_1, int16_t y_1)

  Set the region of interest in cc3_g_pixbuf_frame for virtual windowing.

- bool cc3_pixbuf_frame_set_subsample (cc3_subsample_mode_t mode, uint8_t x_step, uint8_t y_step)

  Set the subsample steps and mode in cc3_g_pixbuf_frame.

- bool cc3_pixbuf_frame_set_coi (cc3_channel_t chan)

  Set the channel of interest for reading from the pixbuf.

- void cc3_pixbuf_frame_reset (void)

  Reset cc3_g_pixbuf_frame to default values.

- void cc3_led_set_state (uint8_t led, bool state)

  Activate or deactivate an LED.

- void cc3_camera_set_power_state (bool state)

  Set the power state of the camera and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_resolution (cc3_camera_resolution_t res)

  Set the resolution of the camera hardware and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_colorspace (cc3_colorspace_t colorspace)

  Set the colorspace of the camera and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_framerate_divider (uint8_t rate_divider)

  Set the framerate divider and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_auto_exposure (bool ae)

  Set auto exposure and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_auto_white_balance (bool wb)

  Set auto white balance and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_brightness (uint8_t level)

  Set the brightness and reset cc3_g_pixbuf_frame.

- void cc3_camera_set_contrast (uint8_t level)

  Set contrast and reset cc3_g_pixbuf_frame.

- bool cc3_camera_set_raw_register (uint8_t address, uint8_t value)

  Using the camera control bus, set a camera register to a value.

- bool cc3_uart_init (uint8_t uart, cc3_uart_rate_t rate, cc3_uart_mode_t mode, cc3_uart_binmode_t binmode)

  Initialize a serial UART.

- uint8_t cc3_uart_get_count (void)

  Get the number of UARTs on this device.

- FILE * cc3_uart_fopen (uint8_t uart, const char *mode)

  Get a file handle for a UART.

- bool cc3_uart_has_data (uint8_t uart)

  Do a non-blocking check to see if data is waiting on the UART.

- uint32_t cc3_timer_get_current_ms (void)

  Get the value of the monotonic timer.

- void cc3_timer_wait_ms (uint32_t delay)

  Wait for a certain amount.

- bool cc3_button_get_state (void)

Get the value of the button right now.

- bool cc3_button_get_and_reset_trigger (void)

  Get and reset the trigger functionality of the button.

- bool cc3_gpio_set_servo_position (uint8_t pin, uint8_t position)

  Set a servo to a position.

- uint8_t cc3_gpio_get_servo_position (uint8_t pin)

  Get a the position of a servo.

- void cc3_gpio_set_mode (uint8_t pin, cc3_gpio_mode_t mode)

  Configure a GPIO pin as input, output, or servo.

- cc3_gpio_mode_t cc3_gpio_get_mode (uint8_t pin)

  Get the current mode setting for a GPIO pin.

- void cc3_gpio_set_value (uint8_t pin, bool value)

  Set a GPIO pin to a value.

- bool cc3_gpio_get_value (uint8_t pin)

  Read the value from a GPIO pin.

- uint8_t cc3_gpio_get_count (void)

  Get the number of available GPIO pins.

Variables

- cc3_frame_t cc3_g_pixbuf_frame

  Current parameters for the internal pixbuf, should be considered read only.

4.1.2 Define Documentation

4.1.2.1 #define CC3_API_MAJOR_VERSION 1

Major version number of API.

This is incremented when backwards-incompatible changes are made in a release.

4.1.2.2 #define CC3_API_MINOR_VERSION 0

Minor version number of API.

This is incremented when compatible new features are added in a release.

4.1.3 Enumeration Type Documentation

4.1.3.1 enum cc3_camera_resolution_t

Allowed resolutions for the camera device.

Enumerator:

CC3_CAMERA_RESOLUTION_LOW Low resolution.
CC3_CAMERA_RESOLUTION_HIGH High resolution.

4.1.3.2 enum cc3_channel_t

Allowable channel values for channel of interest selection and other functions.

Enumerator:

CC3_CHANNEL_SINGLE Only channel in single-channel images.
CC3_CHANNEL_RED Red channel in RGB images.
CC3_CHANNEL_GREEN Green channel in RGB images.
CC3_CHANNEL_BLUE Blue channel in RGB images.
CC3_CHANNEL_Y Y channel in YCrCb images.
CC3_CHANNEL_CR Cr channel in YCrCb images.
CC3_CHANNEL_CB Cb channel in YCrCb images.
CC3_CHANNEL_HUE Hue channel in HSV images.
CC3_CHANNEL_SAT Sat channel in HSV images.
CC3_CHANNEL_VAL Val channel in HSV images.
CC3_CHANNEL_ALL All channels in an image.

4.1.3.3 enum cc3_colorspace_t

Colorspace selector.

Enumerator:

CC3_COLORSPACE_YCRCB YCrCb colorspace.
CC3_COLORSPACE_RGB RGB colorspace.

4.1.3.4 enum cc3_gpio_mode_t

GPIO configuration values.

See also:

cc3_gpio_set_mode() and cc3_gpio_get_mode().

Enumerator:

CC3_GPIO_MODE_INPUT Set pin for input.

CC3_GPIO_MODE_OUTPUT Set pin for output.

CC3_GPIO_MODE_SERVO Set pin for servo output.

4.1.3.5 enum cc3_subsample_mode_t

Subsampling modes.

Enumerator:

CC3_SUBSAMPLE_NEAREST Nearest neighbor subsampling.

CC3_SUBSAMPLE_MEAN Mean of neighbors subsampling.

CC3_SUBSAMPLE_RANDOM Random neighbor subsampling.

4.1.3.6 enum cc3_uart_binmode_t

UART binary/text mode selection values.

See also:

cc3_uart_init() for where to use this.

Enumerator:

CC3_UART_BINMODE_BINARY Do not alter serial data in any way.

CC3_UART_BINMODE_TEXT Read CR as LF, write LF as CR.

4.1.3.7 enum cc3_uart_mode_t

UART modes.

8N1 is the most common.

See also:

cc3_uart_init().

Enumerator:

CC3_UART_MODE_8N1	8 data bits, no parity, 1 stop bit
CC3_UART_MODE_7N1	7 data bits, no parity, 1 stop bit
CC3_UART_MODE_8N2	8 data bits, no parity, 2 stop bits
CC3_UART_MODE_7N2	7 data bits, no parity, 2 stop bits
CC3_UART_MODE_8E1	8 data bits, even parity, 1 stop bit
CC3_UART_MODE_7E1	7 data bits, even parity, 1 stop bit
CC3_UART_MODE_8E2	8 data bits, even parity, 2 stop bits
CC3_UART_MODE_7E2	7 data bits, even parity, 2 stop bits
CC3_UART_MODE_8O1	8 data bits, odd parity, 1 stop bit
CC3_UART_MODE_7O1	7 data bits, odd parity, 1 stop bit
CC3_UART_MODE_8O2	8 data bits, odd parity, 2 stop bits
CC3_UART_MODE_7O2	7 data bits, odd parity, 2 stop bits

4.1.3.8 enum cc3_uart_rate_t

UART speeds.

115200 is the most common.

See also:

cc3_uart_init().

4.1.4 Function Documentation

4.1.4.1 bool cc3_button_get_and_reset_trigger (void)

Get and reset the trigger functionality of the button.

Returns:

true if the button has been pressed since the last time this function was called.

4.1.4.2 bool cc3_button_get_state (void)

Get the value of the button right now.

Returns:

true if button is depressed.

4.1.4.3 bool cc3_camera_init (void)

Initialize camera hardware.

This resets all camera and pixbuf parameters.

Returns:

true if successful.

4.1.4.4 void cc3_camera_set_auto_exposure (bool *ae*)

Set auto exposure and reset cc3_g_pixbuf_frame.

Parameters:

← ***ae*** Auto exposure value.

4.1.4.5 void cc3_camera_set_auto_white_balance (bool *wb*)

Set auto white balance and reset cc3_g_pixbuf_frame.

Parameters:

← ***wb*** Auto white balance value.

4.1.4.6 void cc3_camera_set_brightness (uint8_t *level*)

Set the brightness and reset cc3_g_pixbuf_frame.

Parameters:

← ***level*** The level.

4.1.4.7 void cc3_camera_set_colorspace (cc3_colorspace_t *colorspace*)

Set the colorspace of the camera and reset cc3_g_pixbuf_frame.

Parameters:

← ***colorspace*** The colorspace.

4.1.4.8 void cc3_camera_set_contrast (uint8_t *level*)

Set contrast and reset cc3_g_pixbuf_frame.

Parameters:

← ***level*** The level.

4.1.4.9 void cc3_camera_set_framerate_divider (uint8_t *rate_divider*)

Set the framerate divider and reset cc3_g_pixbuf_frame.

Parameters:

← ***rate_divider*** Desired framerate divider.

4.1.4.10 void cc3_camera_set_power_state (bool *state*)

Set the power state of the camera and reset cc3_g_pixbuf_frame.

Used to conserve power when the camera is not being used.

Note:

If state is set to *false*, the camera will not work.

Parameters:

← ***state*** Set to *true* for normal operation, set to *false* to disable power to the camera.

4.1.4.11 bool cc3_camera_set_raw_register (uint8_t *address*, uint8_t *value*)

Using the camera control bus, set a camera register to a value.

This will take an address and a value from the OmniVision manual and set it on the camera. This should be used for advanced low level manipulation of the camera modes. Currently, this will not set the corresponding cc3 internal data structure that keeps record of the camera mode, nor will it change cc3_g_pixbuf_frame.

Warning:

Use with CAUTION.

Parameters:

← ***address*** The address.

← ***value*** The value.

Returns:

true if successful.

4.1.4.12 void cc3_camera_set_resolution (cc3_camera_resolution_t *res*)

Set the resolution of the camera hardware and reset cc3_g_pixbuf_frame.

Parameters:

← ***res*** Resolution to set.

4.1.4.13 void cc3_filesystem_init (void)

Initialize the filesystem drivers.

Without this call, the FAT filesystem will not be enabled, saving significant code space.

4.1.4.14 uint8_t cc3_gpio_get_count (void)

Get the number of available GPIO pins.

Returns:

The number of GPIO pins.

4.1.4.15 cc3_gpio_mode_t cc3_gpio_get_mode (uint8_t *pin*)

Get the current mode setting for a GPIO pin.

Parameters:

← ***pin*** The pin to assign a mode to.

Returns:

The GPIO mode for this pin.

4.1.4.16 uint8_t cc3_gpio_get_servo_position (uint8_t *pin*)

Get a the position of a servo.

Parameters:

← ***pin*** The pin to query.

Returns:

The servo position value.

4.1.4.17 bool cc3_gpio_get_value (uint8_t *pin*)

Read the value from a GPIO pin.

Note:

If this pin is in input mode, this will return the last value the pin was set to.

Parameters:

← ***pin*** The pin to get.

Returns:

The value of the pin.

4.1.4.18 void cc3_gpio_set_mode (uint8_t *pin*, cc3_gpio_mode_t *mode*)

Configure a GPIO pin as input, output, or servo.

Parameters:

← ***pin*** The pin to set.
← ***mode*** The mode to set the pin to.

4.1.4.19 bool cc3_gpio_set_servo_position (uint8_t *pin*, uint8_t *position*)

Set a servo to a position.

Note:

If the pin is not in servo mode, this function will still set the position, but will not change the mode of the pin.

See also:

cc3_gpio_set_mode() for setting a pin to servo mode.

Parameters:

← ***pin*** The pin to set.

← ***position*** The position to set the servo.

Returns:

true if successful.

4.1.4.20 void cc3_gpio_set_value (uint8_t *pin*, bool *value*)

Set a GPIO pin to a value.

Note:

This has no effect if a pin's mode is set to input. It has only a momentary effect if the pin's mode is set to servo.

Parameters:

← ***pin*** The pin to set.

← ***value*** The value to set the pin to.

4.1.4.21 void cc3_led_set_state (uint8_t *led*, bool *state*)

Activate or deactivate an LED.

Note:

Sometimes a LED is shared with GPIO or other functions.

Parameters:

← ***led*** The LED to act on.

← ***state*** Set to *true* to turn the LED on, *false* to turn the LED off.

4.1.4.22 uint8_t* cc3_malloc_rows (uint32_t *rows*)

Use malloc() to allocate a number of rows of the correct size based on the values in cc3_g_pixbuf_frame.

You must manually use free() to deallocate this memory when you are done.

See also:

cc3_pixbuf_read_rows()

Parameters:

← ***rows*** The number of rows to allocate space for.

Returns:

A pointer to allocated memory or *NULL* if error.

4.1.4.23 bool cc3_pixbuf_frame_set_coi (cc3_channel_t *chan*)

Set the channel of interest for reading from the pixbuf.

Use cc3_channel_t.CC3_ALL for all channels.

Parameters:

← ***chan*** The channel of interest.

Returns:

true on success.

4.1.4.24 bool cc3_pixbuf_frame_set_roi (int16_t *x_0*, int16_t *y_0*, int16_t *x_1*, int16_t *y_1*)

Set the region of interest in cc3_g_pixbuf_frame for virtual windowing.

Parameters:

← ***x_0*** Left horizontal clipping boundary.
← ***y_0*** Top vertical clipping boundary.
← ***x_1*** Right horizontal clipping boundary.
← ***y_1*** Bottom vertical clipping boundary.

Returns:

true if successful.

4.1.4.25 bool cc3_pixbuf_frame_set_subsample (cc3_subsample_mode_t *mode*, uint8_t *x_step*, uint8_t *y_step*)

Set the subsample steps and mode in cc3_g_pixbuf_frame.

Parameters:

← ***mode*** Subsample mode.

← ***x_step*** Horizontal step. Must be either 1 or an even value.

← ***y_step*** Vertical step.

Returns:

true if completely successful. Some settings may be partially applied even if *false*.

4.1.4.26 int cc3_pixbuf_read_rows (void * *mem*, uint32_t *rows*)

Do a row-by-row copy from the pixbuf into a block of memory.

This function takes into account all of the parameters listed in cc3_g_pixbuf_frame.

Parameters:

→ ***mem*** The memory to write the rows to.

← ***rows*** The number of rows to copy.

Returns:

Number of rows copied.

4.1.4.27 void cc3_pixbuf_rewind (void)

Rewind the pixbuf to the beginning without changing the image data stored in it.

This function allows you to seek to the beginning of the pixbuf without calling cc3_pixbuf_load().

4.1.4.28 uint32_t cc3_timer_get_current_ms (void)

Get the value of the monotonic timer.

Returns:

Number of milliseconds since an arbitrary time in the past.

4.1.4.29 void cc3_timer_wait_ms (uint32_t *delay*)

Wait for a certain amount.

Parameters:

← ***delay*** Number of milliseconds to sleep.

4.1.4.30 FILE* cc3_uart_fopen (uint8_t *uart*, const char * *mode*)

Get a file handle for a UART.

Parameters:

← ***uart*** The UART to open.

← ***mode*** Mode as in fopen.

Returns:

The file handle or *NULL* if error.

4.1.4.31 uint8_t cc3_uart_get_count (void)

Get the number of UARTs on this device.

Returns:

Number of UARTs.

4.1.4.32 bool cc3_uart_has_data (uint8_t *uart*)

Do a non-blocking check to see if data is waiting on the UART.

Parameters:

← ***uart*** The UART to check.

Returns:

true if data can be read without blocking.

4.1.4.33 bool cc3_uart_init (uint8_t *uart*, cc3_uart_rate_t *rate*, cc3_uart_mode_t *mode*, cc3_uart_binmode_t *binmode*)

Initialize a serial UART.

Call this for each UART to initialize.

Parameters:

← ***uart*** The UART to initialize.

← ***rate*** The rate.

← ***mode*** The mode.

← ***binmode*** The binary/text mode.

Returns:

true if successful.