The cx2341x driver

Memory at cx2341x chips

This section describes the cx2341x memory map and documents some of the register space.

Note

the memory long words are little-endian ('intel format').

Warning

This information was figured out from searching through the memory and registers, this information may not be correct and is certainly not complete, and was not derived from anything more than searching through the memory space with commands like:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 21)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        ivtvctl -O min=0x02000000,max=0x020000ff
```

So take this as is, I'm always searching for more stuff, it's a large register space :-).

Memory Map

The cx2341x exposes its entire 64M memory space to the PCI host via the PCI BAR0 (Base Address Register 0). The addresses here are offsets relative to the address held in BAR0.

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 35)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        0x00000000-0x00ffffff Encoder memory space
        0x00000000-0x0003ffff Encode.rom
        ???-???         MPEG buffer(s)
        ???-???         Raw video capture buffer(s)
        ???-???         Raw audio capture buffer(s)
        ???-???         Display buffers (6 or 9)

        0x01000000-0x01ffffff Decoder memory space
        0x01000000-0x0103ffff Decode.rom
        ???-???         MPEG buffers(s)
        0x0114b000-0x0115afff Audio.rom (deprecated?)

        0x02000000-0x0200ffff Register Space
```

Registers

The registers occupy the 64k space starting at the 0x02000000 offset from BAR0. All of these registers are 32 bits wide.

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 57)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        DMA Registers 0x000-0xff:
             0x00 - Control:
                     0=reset/cancel, 1=read, 2=write, 4=stop
             0x04 - DMA status:
                     1=read busy, 2=write busy, 4=read error, 8=write error, 16=link list error
             0x08 - pci DMA pointer for read link list
             0x0c - pci DMA pointer for write link list
             0x10 - read/write DMA enable:
                     1=read enable, 2=write enable
             0x14 - always 0xffffffff, if set any lower instability occurs, 0x00 crashes
             0x18 - ??
             0x1c - always 0x20 or 32, smaller values slow down DMA transactions
             0x20 - always value of 0x780a010a
             0x24-0x3c - usually just random values???
             0x40 - Interrupt status
             0x44 - Write a bit here and shows up in Interrupt status 0x40
             0x48 - Interrupt Mask
             0x4C - always value of 0xfffdffff,
                     if changed to 0xffffffff DMA write interrupts break.
             0x50 - always 0xffffffff
             0x54 - always 0xffffffff (0x4c, 0x50, 0x54 seem like interrupt masks, are
                     3 processors on chip, Java ones, VPU, SPU, APU, maybe these are the
                     interrupt masks???).
             0x60-0x7C - random values
             0x80 - first write linked list reg, for Encoder Memory addr
             0x84 - first write linked list reg, for pci memory addr
             0x88 - first write linked list reg, for length of buffer in memory addr
                     (|0x80000000 or this for last link)
             0x8c-0xdc - rest of write linked list reg, 8 sets of 3 total, DMA goes here
                     from linked list addr in reg 0x0c, firmware must push through or
                     something.
             0xe0 - first (and only) read linked list reg, for pci memory addr
             0xe4 - first (and only) read linked list reg, for Decoder memory addr
             0xe8 - first (and only) read linked list reg, for length of buffer
             0xec-0xff - Nothing seems to be in these registers, 0xec-f4 are 0x00000000.
```

Memory locations for Encoder Buffers 0x700-0x7ff:

These registers show offsets of memory locations pertaining to each buffer area used for encoding, have to shift them by <<1 first.

- 0x07F8: Encoder SDRAM refresh
- 0x07FC: Encoder SDRAM pre-charge

Memory locations for Decoder Buffers 0x800-0x8ff:

These registers show offsets of memory locations pertaining to each buffer area used for decoding, have to shift them by <<1 first.

- 0x08F8: Decoder SDRAM refresh
- 0x08FC: Decoder SDRAM pre-charge

Other memory locations:

- 0x2800: Video Display Module control
- 0x2D00: AO (audio output?) control
- 0x2D24: Bytes Flushed
- 0x7000: LSB I2C write clock bit (inverted)
- 0x7004: LSB I2C write data bit (inverted)
- 0x7008: LSB I2C read clock bit
- 0x700c: LSB I2C read data bit
- 0x9008: GPIO get input state
- 0x900c: GPIO set output state
- 0x9020: GPIO direction (Bit7 (GPIO 0..7) - 0:input, 1:output)
- 0x9050: SPU control
- 0x9054: Reset HW blocks
- 0x9058: VPU control
- 0xA018: Bit6: interrupt pending?
- 0xA064: APU command

Interrupt Status Register

The definition of the bits in the interrupt status register 0x0040, and the interrupt mask 0x0048. If a bit is cleared in the mask, then we want our ISR to execute.

- bit 31 Encoder Start Capture
- bit 30 Encoder EOS
- bit 29 Encoder VBI capture

- bit 28 Encoder Video Input Module reset event
- bit 27 Encoder DMA complete
- bit 24 Decoder audio mode change detection event (through event notification)
- bit 22 Decoder data request
- bit 20 Decoder DMA complete
- bit 19 Decoder VBI re-insertion
- bit 18 Decoder DMA err (linked-list bad)

Missing documentation

- Encoder API post(?)
- Decoder API post(?)
- Decoder VTRACE event

The cx2341x firmware upload

This document describes how to upload the cx2341x firmware to the card.

How to find

See the web pages of the various projects that uses this chip for information on how to obtain the firmware.

The firmware stored in a Windows driver can be detected as follows:

- Each firmware image is 256k bytes.
- The 1st 32-bit word of the Encoder image is 0x0000da7
- The 1st 32-bit word of the Decoder image is 0x00003a7
- The 2nd 32-bit word of both images is 0xaa55bb66

How to load

- Issue the FWapi command to stop the encoder if it is running. Wait for the command to complete.
- Issue the FWapi command to stop the decoder if it is running. Wait for the command to complete.
- Issue the I2C command to the digitizer to stop emitting VSYNC events.
- Issue the FWapi command to halt the encoder's firmware.
- Sleep for 10ms.
- Issue the FWapi command to halt the decoder's firmware.
- Sleep for 10ms.
- Write 0x00000000 to register 0x2800 to stop the Video Display Module.
- Write 0x00000005 to register 0x2D00 to stop the AO (audio output?).
- Write 0x00000000 to register 0xA064 to ping? the APU.
- Write 0xFFFFFFFE to register 0x9058 to stop the VPU.
- Write 0xFFFFFFFF to register 0x9054 to reset the HW blocks.
- Write 0x00000001 to register 0x9050 to stop the SPU.
- Sleep for 10ms.
- Write 0x0000001A to register 0x07FC to init the Encoder SDRAM's pre-charge.
- Write 0x80000640 to register 0x07F8 to init the Encoder SDRAM's refresh to 1us.
- Write 0x0000001A to register 0x08FC to init the Decoder SDRAM's pre-charge.
- Write 0x80000640 to register 0x08F8 to init the Decoder SDRAM's refresh to 1us.
- Sleep for 512ms. (600ms is recommended)
- Transfer the encoder's firmware image to offset 0 in Encoder memory space.
- Transfer the decoder's firmware image to offset 0 in Decoder memory space.
- Use a read-modify-write operation to Clear bit 0 of register 0x9050 to re-enable the SPU.
- Sleep for 1 second.
- Use a read-modify-write operation to Clear bits 3 and 0 of register 0x9058 to re-enable the VPU.
- Sleep for 1 second.
- Issue status API commands to both firmware images to verify.

How to call the firmware API

The preferred calling convention is known as the firmware mailbox. The mailboxes are basically a fixed length array that serves as the call-stack.

Firmware mailboxes can be located by searching the encoder and decoder memory for a 16 byte signature. That signature will be located on a 256-byte boundary.

Signature:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 221)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        0x78, 0x56, 0x34, 0x12, 0x12, 0x78, 0x56, 0x34,
        0x34, 0x12, 0x78, 0x56, 0x56, 0x34, 0x12, 0x78
```

The firmware implements 20 mailboxes of 20 32-bit words. The first 10 are reserved for API calls. The second 10 are used by the firmware for event notification.

Index	Name
0	Flags
1	Command
2	Return value
3	Timeout
4-19	Parameter/Result

The flags are defined in the following table. The direction is from the perspective of the firmware.

Bit	Direction	Purpose
2	O	Firmware has processed the command.
1	I	Driver has finished setting the parameters.
0	I	Driver is using this mailbox.

The command is a 32-bit enumerator. The API specifics may be found in this chapter.

The return value is a 32-bit enumerator. Only two values are currently defined:

- 0=success
- -1=command undefined.

There are 16 parameters/results 32-bit fields. The driver populates these fields with values for all the parameters required by the call. The driver overwrites these fields with result values returned by the call.

The timeout value protects the card from a hung driver thread. If the driver doesn't handle the completed call within the timeout specified, the firmware will reset that mailbox.

To make an API call, the driver iterates over each mailbox looking for the first one available (bit 0 has been cleared). The driver sets that bit, fills in the command enumerator, the timeout value and any required parameters. The driver then sets the parameter ready bit (bit 1). The firmware scans the mailboxes for pending commands, processes them, sets the result code, populates the result value array with that call's return values and sets the call complete bit (bit 2). Once bit 2 is set, the driver should retrieve the results and clear all the flags. If the driver does not perform this task within the time set in the timeout register, the firmware will reset that mailbox.

Event notifications are sent from the firmware to the host. The host tells the firmware which events it is interested in via an API call. That call tells the firmware which notification mailbox to use. The firmware signals the host via an interrupt. Only the 16 Results fields are used, the Flags, Command, Return value and Timeout words are not used.

OSD firmware API description

Note

this API is part of the decoder firmware, so it's cx23415 only.

CX2341X_OSD_GET_FRAMEBUFFER

Enum: 65/0x41

Description

Return base and length of contiguous OSD memory.

Result[0]

OSD base address

Result[1]

OSD length

CX2341X_OSD_GET_PIXEL_FORMAT

Enum: 66/0x42

Description

Query OSD format

Result[0]

0=8bit index 1=16bit RGB 5:6:5 2=16bit ARGB 1:5:5:5 3=16bit ARGB 1:4:4:4 4=32bit ARGB 8:8:8:8

CX2341X_OSD_SET_PIXEL_FORMAT

Enum: 67/0x43

Description

Assign pixel format

Param[0]

- 0=8bit index
- 1=16bit RGB 5:6:5
- 2=16bit ARGB 1:5:5:5
- 3=16bit ARGB 1:4:4:4
- 4=32bit ARGB 8:8:8:8

CX2341X_OSD_GET_STATE

Enum: 68/0x44

Description

Query OSD state

Result[0]

- Bit 0 0=off, 1=on
- Bits 1:2 alpha control
- Bits 3:5 pixel format

CX2341X_OSD_SET_STATE

Enum: 69/0x45

Description

OSD switch

Param[0]

0=off, 1=on

CX2341X_OSD_GET_OSD_COORDS

Enum: 70/0x46

Description

Retrieve coordinates of OSD area blended with video

Result[0]

OSD buffer address

Result[1]

Stride in pixels

Result[2]

Lines in OSD buffer

Result[3]

Horizontal offset in buffer

Result[4]

Vertical offset in buffer

CX2341X_OSD_SET_OSD_COORDS

Enum: 71/0x47

Description

Assign the coordinates of the OSD area to blend with video

Param[0]

buffer address

Param[1]

buffer stride in pixels

Param[2]

lines in buffer

Param[3]

horizontal offset

Param[4]

vertical offset

CX2341X_OSD_GET_SCREEN_COORDS

Enum: 72/0x48

Description

Retrieve OSD screen area coordinates

Result[0]

top left horizontal offset

Result[1]

top left vertical offset

Result[2]

bottom right horizontal offset

Result[3]

bottom right vertical offset

CX2341X_OSD_SET_SCREEN_COORDS

Enum: 73/0x49

Description

Assign the coordinates of the screen area to blend with video

Param[0]

top left horizontal offset

Param[1]

top left vertical offset

Param[2]

bottom left horizontal offset

Param[3]

bottom left vertical offset

CX2341X_OSD_GET_GLOBAL_ALPHA

Enum: 74/0x4A

Description

Retrieve OSD global alpha

Result[0]

global alpha: 0=off, 1=on

Result[1]

bits 0:7 global alpha

CX2341X_OSD_SET_GLOBAL_ALPHA

Enum: 75/0x4B

Description

Update global alpha

Param[0]

global alpha: 0=off, 1=on

Param[1]

global alpha (8 bits)

Param[2]

local alpha: 0=on, 1=off

CX2341X_OSD_SET_BLEND_COORDS

Enum: 78/0x4C

Description

Move start of blending area within display buffer

Param[0]

horizontal offset in buffer

Param[1]

vertical offset in buffer

CX2341X_OSD_GET_FLICKER_STATE

Enum: 79/0x4F

Description

Retrieve flicker reduction module state

Result[0]

flicker state: 0=off, 1=on

CX2341X_OSD_SET_FLICKER_STATE

Enum: 80/0x50

Description

Set flicker reduction module state

Param[0]

State: 0=off, 1=on

CX2341X_OSD_BLT_COPY

Enum: 82/0x52

Description

BLT copy

Param[0]

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 648)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        '0000'  zero
        '0001' ~destination AND ~source
        '0010' ~destination AND  source
        '0011' ~destination
        '0100'  destination AND ~source
        '0101'                  ~source
        '0110'  destination XOR  source
        '0111' ~destination OR  ~source
        '1000' ~destination AND ~source
        '1001'  destination XNOR source
        '1010'                   source
        '1011' ~destination OR   source
        '1100'  destination
        '1101'  destination OR  ~source
        '1110'  destination OR   source
        '1111'  one
```

Param[1]

Resulting alpha blending

- '01' source_alpha
- '10' destination_alpha
- '11' source_alpha*destination_alpha+1 (zero if both source and destination alpha are zero)

Param[2]

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 681)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        '00' output_pixel = source_pixel

        '01' if source_alpha=0:
                 output_pixel = destination_pixel
             if 256 > source_alpha > 1:
                 output_pixel = ((source_alpha + 1)*source_pixel +
                                 (255 - source_alpha)*destination_pixel)/256

        '10' if destination_alpha=0:
```

```
                    output_pixel = source_pixel
                 if 255 > destination_alpha > 0:
                    output_pixel = ((255 - destination_alpha)*source_pixel +
                                    (destination_alpha + 1)*destination_pixel)/256

           '11' if source_alpha=0:
                    source_temp = 0
                if source_alpha=255:
                    source_temp = source_pixel*256
                if 255 > source_alpha > 0:
                    source_temp = source_pixel*(source_alpha + 1)
                if destination_alpha=0:
                    destination_temp = 0
                if destination_alpha=255:
                    destination_temp = destination_pixel*256
                if 255 > destination_alpha > 0:
                    destination_temp = destination_pixel*(destination_alpha + 1)
                output_pixel = (source_temp + destination_temp)/256
```

Param[3]

width

Param[4]

height

Param[5]

destination pixel mask

Param[6]

destination rectangle start address

Param[7]

destination stride in dwords

Param[8]

source stride in dwords

Param[9]

source rectangle start address

CX2341X_OSD_BLT_FILL

Enum: 83/0x53

Description

BLT fill color

Param[0]

Same as Param[0] on API 0x52

Param[1]

Same as Param[1] on API 0x52

Param[2]

Same as Param[2] on API 0x52

Param[3]

width

Param[4]

height

Param[5]

destination pixel mask

Param[6]

destination rectangle start address

Param[7]

destination stride in dwords

Param[8]

color fill value

CX2341X_OSD_BLT_TEXT

Enum: 84/0x54

Description

BLT for 8 bit alpha text source

Param[0]

Same as Param[0] on API 0x52

Param[1]

Same as Param[1] on API 0x52

Param[2]

Same as Param[2] on API 0x52

Param[3]

width

Param[4]

height

Param[5]

destination pixel mask

Param[6]

destination rectangle start address

Param[7]

destination stride in dwords

Param[8]

source stride in dwords

Param[9]

source rectangle start address

Param[10]

color fill value

CX2341X_OSD_SET_FRAMEBUFFER_WINDOW

Enum: 86/0x56

Description

Positions the main output window on the screen. The coordinates must be such that the entire window fits on the screen.

Param[0]

window width

Param[1]

window height

Param[2]

top left window corner horizontal offset

Param[3]

top left window corner vertical offset

CX2341X_OSD_SET_CHROMA_KEY

Enum: 96/0x60

Description

Chroma key switch and color

Param[0]

state: 0=off, 1=on

Param[1]

color

CX2341X_OSD_GET_ALPHA_CONTENT_INDEX

Enum: 97/0x61

Description

Retrieve alpha content index

Result[0]

alpha content index, Range 0:15

CX2341X_OSD_SET_ALPHA_CONTENT_INDEX

Enum: 98/0x62

Description

Assign alpha content index

Param[0]

alpha content index, range 0:15

Encoder firmware API description

CX2341X_ENC_PING_FW

Enum: 128/0x80

Description

Does nothing. Can be used to check if the firmware is responding.

CX2341X_ENC_START_CAPTURE

Enum: 129/0x81

Description

Commences the capture of video, audio and/or VBI data. All encoding parameters must be initialized prior to this API call. Captures frames continuously or until a predefined number of frames have been captured.

Param[0]

Capture stream type:

- 0=MPEG
- 1=Raw
- 2=Raw passthrough
- 3=VBI

Param[1]

Bitmask:

- Bit 0 when set, captures YUV
- Bit 1 when set, captures PCM audio
- Bit 2 when set, captures VBI (same as param[0]=3)
- Bit 3 when set, the capture destination is the decoder (same as param[0]=2)
- Bit 4 when set, the capture destination is the host

> **Note**
>
> this parameter is only meaningful for RAW capture type.

CX2341X_ENC_STOP_CAPTURE

Enum: 130/0x82

Description

Ends a capture in progress

Param[0]

- 0=stop at end of GOP (generates IRQ)
- 1=stop immediate (no IRQ)

Param[1]

Stream type to stop, see param[0] of API 0x81

Param[2]

Subtype, see param[1] of API 0x81

CX2341X_ENC_SET_AUDIO_ID

Enum: 137/0x89

Description

Assigns the transport stream ID of the encoded audio stream

Param[0]

Audio Stream ID

CX2341X_ENC_SET_VIDEO_ID

Enum: 139/0x8B

Description

Set video transport stream ID

Param[0]

Video stream ID

CX2341X_ENC_SET_PCR_ID

Enum: 141/0x8D

Description

Assigns the transport stream ID for PCR packets

Param[0]

PCR Stream ID

CX2341X_ENC_SET_FRAME_RATE

Enum: 143/0x8F

Description

Set video frames per second. Change occurs at start of new GOP.

Param[0]

- 0=30fps
- 1=25fps

CX2341X_ENC_SET_FRAME_SIZE

Enum: 145/0x91

Description

Select video stream encoding resolution.

Param[0]

Height in lines. Default 480

Param[1]

Width in pixels. Default 720

CX2341X_ENC_SET_BIT_RATE

Enum: 149/0x95

Description

Assign average video stream bitrate.

Param[0]

0=variable bitrate, 1=constant bitrate

Param[1]

bitrate in bits per second

Param[2]

peak bitrate in bits per second, divided by 400

Param[3]

Mux bitrate in bits per second, divided by 400. May be 0 (default).

Param[4]

Rate Control VBR Padding

Param[5]

VBV Buffer used by encoder

> **Note**
>
> 1. Param[3] and Param[4] seem to be always 0
> 2. Param[5] doesn't seem to be used.

CX2341X_ENC_SET_GOP_PROPERTIES

Enum: 151/0x97

Description

Setup the GOP structure

Param[0]

GOP size (maximum is 34)

Param[1]

Number of B frames between the I and P frame, plus 1. For example: IBBPBBPBBPBB --> GOP size: 12, number of B frames: 2+1 = 3

> **Note**
>
> GOP size must be a multiple of (B-frames + 1).

CX2341X_ENC_SET_ASPECT_RATIO

Enum: 153/0x99

Description

Sets the encoding aspect ratio. Changes in the aspect ratio take effect at the start of the next GOP.

Param[0]

- '0000' forbidden
- '0001' 1:1 square
- '0010' 4:3
- '0011' 16:9
- '0100' 2.21:1
- '0101' to '1111' reserved

CX2341X_ENC_SET_DNR_FILTER_MODE

Enum: 155/0x9B

Description

Assign Dynamic Noise Reduction operating mode

Param[0]

Bit0: Spatial filter, set=auto, clear=manual Bit1: Temporal filter, set=auto, clear=manual

Param[1]

Median filter:

- 0=Disabled
- 1=Horizontal
- 2=Vertical
- 3=Horiz/Vert
- 4=Diagonal

CX2341X_ENC_SET_DNR_FILTER_PROPS

Enum: 157/0x9D

Description

These Dynamic Noise Reduction filter values are only meaningful when the respective filter is set to "manual" (See API 0x9B)

Param[0]

Spatial filter: default 0, range 0:15

Param[1]

Temporal filter: default 0, range 0:31

CX2341X_ENC_SET_CORING_LEVELS

Enum: 159/0x9F

Description

Assign Dynamic Noise Reduction median filter properties.

Param[0]

Threshold above which the luminance median filter is enabled. Default: 0, range 0:255

Param[1]

Threshold below which the luminance median filter is enabled. Default: 255, range 0:255

Param[2]

Threshold above which the chrominance median filter is enabled. Default: 0, range 0:255

Param[3]

Threshold below which the chrominance median filter is enabled. Default: 255, range 0:255

CX2341X_ENC_SET_SPATIAL_FILTER_TYPE

Enum: 161/0xA1

Description

Assign spatial prefilter parameters

Param[0]

Luminance filter

- 0=Off
- 1=1D Horizontal
- 2=1D Vertical
- 3=2D H/V Separable (default)
- 4=2D Symmetric non-separable

Param[1]

Chrominance filter

- 0=Off
- 1=1D Horizontal (default)

CX2341X_ENC_SET_VBI_LINE

Enum: 183/0xB7

Description

Selects VBI line number.

Param[0]

- Bits 0:4 line number
- Bit 31 0=top_field, 1=bottom_field
- Bits 0:31 all set specifies "all lines"

Param[1]

VBI line information features: 0=disabled, 1=enabled

Param[2]

Slicing: 0=None, 1=Closed Caption Almost certainly not implemented. Set to 0.

Param[3]

Luminance samples in this line. Almost certainly not implemented. Set to 0.

Param[4]

Chrominance samples in this line Almost certainly not implemented. Set to 0.

CX2341X_ENC_SET_STREAM_TYPE

Enum: 185/0xB9

Description

Assign stream type

> **Note**
>
> Transport stream is not working in recent firmwares. And in older firmwares the timestamps in the TS seem to be unreliable.

Param[0]

- 0=Program stream
- 1=Transport stream
- 2=MPEG1 stream
- 3=PES A/V stream
- 5=PES Video stream
- 7=PES Audio stream
- 10=DVD stream
- 11=VCD stream
- 12=SVCD stream
- 13=DVD_S1 stream
- 14=DVD_S2 stream

CX2341X_ENC_SET_OUTPUT_PORT

Enum: 187/0xBB

Description

Assign stream output port. Normally 0 when the data is copied through the PCI bus (DMA), and 1 when the data is streamed to another chip (pvrusb and cx88-blackbird).

Param[0]

- 0=Memory (default)
- 1=Streaming
- 2=Serial

Param[1]

Unknown, but leaving this to 0 seems to work best. Indications are that this might have to do with USB support, although passing anything but 0 only breaks things.

CX2341X_ENC_SET_AUDIO_PROPERTIES

Enum: 189/0xBD

Description

Set audio stream properties, may be called while encoding is in progress.

> **Note**
>
> All bitfields are consistent with ISO11172 documentation except bits 2:3 which ISO docs define as:
>
> - '11' Layer I
> - '10' Layer II
> - '01' Layer III
> - '00' Undefined
>
> This discrepancy may indicate a possible error in the documentation. Testing indicated that only Layer II is actually working, and that the minimum bitrate should be 192 kbps.

Param[0]

Bitmask:

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst, line 1481)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

           0:1  '00' 44.1Khz
                '01' 48Khz
                '10' 32Khz
                '11' reserved

           2:3  '01'=Layer I
                '10'=Layer II

           4:7  Bitrate:
                     Index | Layer I     | Layer II
                     ------+-------------+------------
                    '0000' | free format | free format
                    '0001' |  32 kbit/s  |  32 kbit/s
                    '0010' |  64 kbit/s  |  48 kbit/s
                    '0011' |  96 kbit/s  |  56 kbit/s
                    '0100' | 128 kbit/s  |  64 kbit/s
                    '0101' | 160 kbit/s  |  80 kbit/s
                    '0110' | 192 kbit/s  |  96 kbit/s
                    '0111' | 224 kbit/s  | 112 kbit/s
                    '1000' | 256 kbit/s  | 128 kbit/s
                    '1001' | 288 kbit/s  | 160 kbit/s
                    '1010' | 320 kbit/s  | 192 kbit/s
                    '1011' | 352 kbit/s  | 224 kbit/s
                    '1100' | 384 kbit/s  | 256 kbit/s
                    '1101' | 416 kbit/s  | 320 kbit/s
                    '1110' | 448 kbit/s  | 384 kbit/s

                .. note::

                        For Layer II, not all combinations of total bitrate
                        and mode are allowed. See ISO11172-3 3-Annex B,
                        Table 3-B.2

           8:9  '00'=Stereo
                '01'=JointStereo
                '10'=Dual
                '11'=Mono

                .. note::

                        The cx23415 cannot decode Joint Stereo properly.

          10:11 Mode Extension used in joint_stereo mode.
                In Layer I and II they indicate which subbands are in
                intensity_stereo. All other subbands are coded in stereo.
                    '00' subbands 4-31 in intensity_stereo, bound==4
                    '01' subbands 8-31 in intensity_stereo, bound==8
                    '10' subbands 12-31 in intensity_stereo, bound==12
                    '11' subbands 16-31 in intensity_stereo, bound==16

          12:13 Emphasis:
                    '00' None
                    '01' 50/15uS
                    '10' reserved
                    '11' CCITT J.17

          14    CRC:
                    '0' off
                    '1' on

          15    Copyright:
                    '0' off
                    '1' on

          16    Generation:
                    '0' copy
                    '1' original
```

CX2341X_ENC_HALT_FW

Enum: 195/0xC3

Description

The firmware is halted and no further API calls are serviced until the firmware is uploaded again.

CX2341X_ENC_GET_VERSION

Enum: 196/0xC4

Description

Returns the version of the encoder firmware.

Result[0]

Version bitmask: - Bits 0:15 build - Bits 16:23 minor - Bits 24:31 major

CX2341X_ENC_SET_GOP_CLOSURE

Enum: 197/0xC5

Description

Assigns the GOP open/close property.

Param[0]

- 0=Open
- 1=Closed

CX2341X_ENC_GET_SEQ_END

Enum: 198/0xC6

Description

Obtains the sequence end code of the encoder's buffer. When a capture is started a number of interrupts are still generated, the last of which will have Result[0] set to 1 and Result[1] will contain the size of the buffer.

Result[0]

State of the transfer (1 if last buffer)

Result[1]

If Result[0] is 1, this contains the size of the last buffer, undefined otherwise.

CX2341X_ENC_SET_PGM_INDEX_INFO

Enum: 199/0xC7

Description

Sets the Program Index Information. The information is stored as follows:

```
struct info {
        u32 length;             // Length of this frame
        u32 offset_low;         // Offset in the file of the
        u32 offset_high;        // start of this frame
        u32 mask1;              // Bits 0-2 are the type mask:
                                // 1=I, 2=P, 4=B
                                // 0=End of Program Index, other fields
                                //   are invalid.
        u32 pts;                // The PTS of the frame
        u32 mask2;              // Bit 0 is bit 32 of the pts.
};
u32 table_ptr;
struct info index[400];
```

The table_ptr is the encoder memory address in the table were *new* entries will be written.

> **Note**
>
> This is a ringbuffer, so the table_ptr will wraparound.

Param[0]

Picture Mask: - 0=No index capture - 1=I frames - 3=I,P frames - 7=I,P,B frames

(Seems to be ignored, it always indexes I, P and B frames)

Param[1]

Elements requested (up to 400)

Result[0]

Offset in the encoder memory of the start of the table.

Result[1]

Number of allocated elements up to a maximum of Param[1]

CX2341X_ENC_SET_VBI_CONFIG

Enum: 200/0xC8

Description

Configure VBI settings

Param[0]

Bitmap:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 1705)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

            0    Mode '0' Sliced, '1' Raw
            1:3  Insertion:
                     '000' insert in extension & user data
                     '001' insert in private packets
                     '010' separate stream and user data
                     '111' separate stream and private data
            8:15 Stream ID (normally 0xBD)
```

Param[1]

Frames per interrupt (max 8). Only valid in raw mode.

Param[2]

Total raw VBI frames. Only valid in raw mode.

Param[3]

Start codes

Param[4]

Stop codes

Param[5]

Lines per frame

Param[6]

Byte per line

Result[0]

Observed frames per interrupt in raw mode only. Rage 1 to Param[1]

Result[1]

Observed number of frames in raw mode. Range 1 to Param[2]

Result[2]

Memory offset to start or raw VBI data

CX2341X_ENC_SET_DMA_BLOCK_SIZE

Enum: 201/0xC9

Description

Set DMA transfer block size

Param[0]

DMA transfer block size in bytes or frames. When unit is bytes, supported block sizes are 2^7, 2^8 and 2^9 bytes.

Param[1]

Unit: 0=bytes, 1=frames

CX2341X_ENC_GET_PREV_DMA_INFO_MB_10

Enum: 202/0xCA

Description

Returns information on the previous DMA transfer in conjunction with bit 27 of the interrupt mask. Uses mailbox 10.

Result[0]

Type of stream

Result[1]

Address Offset

Result[2]

Maximum size of transfer

CX2341X_ENC_GET_PREV_DMA_INFO_MB_9

Enum: 203/0xCB

Description

Returns information on the previous DMA transfer in conjunction with bit 27 or 18 of the interrupt mask. Uses mailbox 9.

Result[0]

Status bits: - 0 read completed - 1 write completed - 2 DMA read error - 3 DMA write error - 4 Scatter-Gather array error

Result[1]

DMA type

Result[2]

Presentation Time Stamp bits 0..31

Result[3]

Presentation Time Stamp bit 32

CX2341X_ENC_SCHED_DMA_TO_HOST

Enum: 204/0xCC

Description

Setup DMA to host operation

Param[0]

Memory address of link list

Param[1]

Length of link list (wtf: what units ???)

Param[2]

DMA type (0=MPEG)

CX2341X_ENC_INITIALIZE_INPUT

Enum: 205/0xCD

Description

Initializes the video input

CX2341X_ENC_SET_FRAME_DROP_RATE

Enum: 208/0xD0

Description

For each frame captured, skip specified number of frames.

Param[0]

Number of frames to skip

CX2341X_ENC_PAUSE_ENCODER

Enum: 210/0xD2

Description

During a pause condition, all frames are dropped instead of being encoded.

Param[0]

- 0=Pause encoding
- 1=Continue encoding

CX2341X_ENC_REFRESH_INPUT

Enum: 211/0xD3

Description

Refreshes the video input

CX2341X_ENC_SET_COPYRIGHT

Enum: 212/0xD4

Description

Sets stream copyright property

Param[0]

- 0=Stream is not copyrighted
- 1=Stream is copyrighted

CX2341X_ENC_SET_EVENT_NOTIFICATION

Enum: 213/0xD5

Description

Setup firmware to notify the host about a particular event. Host must unmask the interrupt bit.

Param[0]

Event (0=refresh encoder input)

Param[1]

Notification 0=disabled 1=enabled

Param[2]

Interrupt bit

Param[3]

Mailbox slot, -1 if no mailbox required.

CX2341X_ENC_SET_NUM_VSYNC_LINES

Enum: 214/0xD6

Description

Depending on the analog video decoder used, this assigns the number of lines for field 1 and 2.

Param[0]

Field 1 number of lines: - 0x00EF for SAA7114 - 0x00F0 for SAA7115 - 0x0105 for Micronas

Param[1]

Field 2 number of lines: - 0x00EF for SAA7114 - 0x00F0 for SAA7115 - 0x0106 for Micronas

CX2341X_ENC_SET_PLACEHOLDER

Enum: 215/0xD7

Description

Provides a mechanism of inserting custom user data in the MPEG stream.

Param[0]

- 0=extension & user data
- 1=private packet with stream ID 0xBD

Param[1]

Rate at which to insert data, in units of frames (for private packet) or GOPs (for ext. & user data)

Param[2]

Number of data DWORDs (below) to insert

Param[3]

Custom data 0

Param[4]

Custom data 1

Param[5]

Custom data 2

Param[6]

Custom data 3

Param[7]

Custom data 4

Param[8]

Custom data 5

Param[9]

Custom data 6

Param[10]

Custom data 7

Param[11]

Custom data 8

CX2341X_ENC_MUTE_VIDEO

Enum: 217/0xD9

Description

Video muting

Param[0]

Bit usage:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 2107)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

         0      '0'=video not muted
                '1'=video muted, creates frames with the YUV color defined below
         1:7    Unused
         8:15   V chrominance information
        16:23   U chrominance information
        24:31   Y luminance information
```

CX2341X_ENC_MUTE_AUDIO

Enum: 218/0xDA

Description

Audio muting

Param[0]

- 0=audio not muted
- 1=audio muted (produces silent mpeg audio stream)

CX2341X_ENC_SET_VERT_CROP_LINE

Enum: 219/0xDB

Description

Something to do with 'Vertical Crop Line'

Param[0]

If saa7114 and raw VBI capture and 60 Hz, then set to 10001. Else 0.

CX2341X_ENC_MISC

CX2341X_ENC_MISC

Enum: 220/0xDC

Description

Miscellaneous actions. Not known for 100% what it does. It's really a sort of ioctl call. The first parameter is a command number, the second the value.

Param[0]

Command number:

> **System Message: WARNING/2** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master)(Documentation)(driver-api)(media)(drivers)cx2341x-devel.rst`, **line 2171**)
>
> Cannot analyze code. No Pygments lexer found for "none".
>
> ```
> .. code-block:: none
>
>          1=set initial SCR value when starting encoding (works).
>          2=set quality mode (apparently some test setting).
>          3=setup advanced VIM protection handling.
>            Always 1 for the cx23416 and 0 for cx23415.
>          4=generate DVD compatible PTS timestamps
>          5=USB flush mode
>          6=something to do with the quantization matrix
>          7=set navigation pack insertion for DVD: adds 0xbf (private stream 2)
>            packets to the MPEG. The size of these packets is 2048 bytes (including
>            the header of 6 bytes: 0x000001bf + length). The payload is zeroed and
>            it is up to the application to fill them in. These packets are apparently
>            inserted every four frames.
>          8=enable scene change detection (seems to be a failure)
>          9=set history parameters of the video input module
>         10=set input field order of VIM
>         11=set quantization matrix
>         12=reset audio interface after channel change or input switch (has no argument).
>            Needed for the cx2584x, not needed for the mspx4xx, but it doesn't seem to
>            do any harm calling it regardless.
>         13=set audio volume delay
>         14=set audio delay
> ```

Param[1]

Command value.

Decoder firmware API description

> **Note**
>
> this API is part of the decoder firmware, so it's cx23415 only.

CX2341X_DEC_PING_FW

Enum: 0/0x00

Description

This API call does nothing. It may be used to check if the firmware is responding.

CX2341X_DEC_START_PLAYBACK

Enum: 1/0x01

Description

Begin or resume playback.

Param[0]

0 based frame number in GOP to begin playback from.

Param[1]

Specifies the number of muted audio frames to play before normal audio resumes. (This is not implemented in the firmware, leave at 0)

CX2341X_DEC_STOP_PLAYBACK

Enum: 2/0x02

Description

Ends playback and clears all decoder buffers. If PTS is not zero, playback stops at specified PTS.

Param[0]

Display 0=last frame, 1=black

> **Note**
>
> this takes effect immediately, so if you want to wait for a PTS, then use '0', otherwise the screen goes to black at once. You can call this later (even if there is no playback) with a 1 value to set the screen to black.

Param[1]

PTS low

Param[2]

PTS high

CX2341X_DEC_SET_PLAYBACK_SPEED

Enum: 3/0x03

Description

Playback stream at speed other than normal. There are two modes of operation:

- Smooth: host transfers entire stream and firmware drops unused frames.
- Coarse: host drops frames based on indexing as required to achieve desired speed.

Param[0]

> **System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 2298)**
>
> Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        Bitmap:
            0:7  0 normal
                 1 fast only "1.5 times"
                 n nX fast, 1/nX slow
            30   Framedrop:
                     '0' during 1.5 times play, every other B frame is dropped
                     '1' during 1.5 times play, stream is unchanged (bitrate
                         must not exceed 8mbps)
            31   Speed:
                     '0' slow
                     '1' fast
```

> **Note**
>
> n is limited to 2. Anything higher does not result in faster playback. Instead the host should start dropping frames.

Param[1]

Direction: 0=forward, 1=reverse

> **Note**

to make reverse playback work you have to write full GOPs in reverse order.

Param[2]

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master) (Documentation) (driver-api) (media) (drivers) cx2341x-devel.rst`, line 2330)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        Picture mask:
            1=I frames
            3=I, P frames
            7=I, P, B frames
```

Param[3]

B frames per GOP (for reverse play only)

Note

for reverse playback the Picture Mask should be set to I or I, P. Adding B frames to the mask will result in corrupt video. This field has to be set to the correct value in order to keep the timing correct.

Param[4]

Mute audio: 0=disable, 1=enable

Param[5]

Display 0=frame, 1=field

Param[6]

Specifies the number of muted audio frames to play before normal audio resumes. (Not implemented in the firmware, leave at 0)

CX2341X_DEC_STEP_VIDEO

Enum: 5/0x05

Description

Each call to this API steps the playback to the next unit defined below in the current playback direction.

Param[0]

0=frame, 1=top field, 2=bottom field

CX2341X_DEC_SET_DMA_BLOCK_SIZE

Enum: 8/0x08

Description

Set DMA transfer block size. Counterpart to API 0xC9

Param[0]

DMA transfer block size in bytes. A different size may be specified when issuing the DMA transfer command.

CX2341X_DEC_GET_XFER_INFO

Enum: 9/0x09

Description

This API call may be used to detect an end of stream condition.

Result[0]

Stream type

Result[1]

Address offset

Result[2]

Maximum bytes to transfer

Result[3]

Buffer fullness

CX2341X_DEC_GET_DMA_STATUS

Enum: 10/0x0A

Description

Status of the last DMA transfer

Result[0]

Bit 1 set means transfer complete Bit 2 set means DMA error Bit 3 set means linked list error

Result[1]

DMA type: 0=MPEG, 1=OSD, 2=YUV

CX2341X_DEC_SCHED_DMA_FROM_HOST

Enum: 11/0x0B

Description

Setup DMA from host operation. Counterpart to API 0xCC

Param[0]

Memory address of link list

Param[1]

Total # of bytes to transfer

Param[2]

DMA type (0=MPEG, 1=OSD, 2=YUV)

CX2341X_DEC_PAUSE_PLAYBACK

Enum: 13/0x0D

Description

Freeze playback immediately. In this mode, when internal buffers are full, no more data will be accepted and data request IRQs will be masked.

Param[0]

Display: 0=last frame, 1=black

CX2341X_DEC_HALT_FW

Enum: 14/0x0E

Description

The firmware is halted and no further API calls are serviced until the firmware is uploaded again.

CX2341X_DEC_SET_STANDARD

Enum: 16/0x10

Description

Selects display standard

Param[0]

0=NTSC, 1=PAL

CX2341X_DEC_GET_VERSION

Enum: 17/0x11

Description

Returns decoder firmware version information

Result[0]

Version bitmask:

- Bits 0:15 build
- Bits 16:23 minor
- Bits 24:31 major

CX2341X_DEC_SET_STREAM_INPUT

Enum: 20/0x14

Description

Select decoder stream input port

Param[0]

0=memory (default), 1=streaming

CX2341X_DEC_GET_TIMING_INFO

Enum: 21/0x15

Description

Returns timing information from start of playback

Result[0]

Frame count by decode order

Result[1]

Video PTS bits 0:31 by display order

Result[2]

Video PTS bit 32 by display order

Result[3]

SCR bits 0:31 by display order

Result[4]

SCR bit 32 by display order

CX2341X_DEC_SET_AUDIO_MODE

Enum: 22/0x16

Description

Select audio mode

Param[0]

Dual mono mode action

    0=Stereo, 1=Left, 2=Right, 3=Mono, 4=Swap, -1=Unchanged

Param[1]

Stereo mode action:
0=Stereo, 1=Left, 2=Right, 3=Mono, 4=Swap, -1=Unchanged

CX2341X_DEC_SET_EVENT_NOTIFICATION

Enum: 23/0x17

Description

Setup firmware to notify the host about a particular event. Counterpart to API 0xD5

Param[0]

Event:

- 0=Audio mode change between mono, (joint) stereo and dual channel.
- 3=Decoder started
- 4=Unknown: goes off 10-15 times per second while decoding.
- 5=Some sync event: goes off once per frame.

Param[1]

Notification 0=disabled, 1=enabled

Param[2]

Interrupt bit

Param[3]

Mailbox slot, -1 if no mailbox required.

CX2341X_DEC_SET_DISPLAY_BUFFERS

Enum: 24/0x18

Description

Number of display buffers. To decode all frames in reverse playback you must use nine buffers.

Param[0]

0=six buffers, 1=nine buffers

CX2341X_DEC_EXTRACT_VBI

Enum: 25/0x19

Description

Extracts VBI data

Param[0]

0=extract from extension & user data, 1=extract from private packets

Result[0]

VBI table location

Result[1]

VBI table size

CX2341X_DEC_SET_DECODER_SOURCE

Enum: 26/0x1A

Description

Selects decoder source. Ensure that the parameters passed to this API match the encoder settings.

Param[0]

Mode: 0=MPEG from host, 1=YUV from encoder, 2=YUV from host

Param[1]

YUV picture width

Param[2]

YUV picture height

Param[3]

Bitmap: see Param[0] of API 0xBD

CX2341X_DEC_SET_PREBUFFERING

Enum: 30/0x1E

Description

Decoder prebuffering, when enabled up to 128KB are buffered for streams <8mpbs or 640KB for streams >8mbps

Param[0]

0=off, 1=on

PVR350 Video decoder registers 0x02002800 -> 0x02002B00

Author: Ian Armstrong <ian@iarmst.demon.co.uk>

Version: v0.4

Date: 12 March 2007

This list has been worked out through trial and error. There will be mistakes and omissions. Some registers have no obvious effect so it's hard to say what they do, while others interact with each other, or require a certain load sequence. Horizontal filter setup is one example, with six registers working in unison and requiring a certain load sequence to correctly configure. The indexed colour palette is much easier to set at just two registers, but again it requires a certain load sequence.

Some registers are fussy about what they are set to. Load in a bad value & the decoder will fail. A firmware reload will often recover, but sometimes a reset is required. For registers containing size information, setting them to 0 is generally a bad idea. For other control registers i.e. 2878, you'll only find out what values are bad when it hangs.

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\drivers\(linux-master)(Documentation)(driver-api)(media)(drivers)cx2341x-devel.rst`, line 2787)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

        --------------------------------------------------------------------------------
        2800
        bit 0
                Decoder enable
                0 = disable
                1 = enable
        --------------------------------------------------------------------------------
        2804
        bits 0:31
                Decoder horizontal Y alias register 1
        ---------------
        2808
        bits 0:31
                Decoder horizontal Y alias register 2
        ---------------
        280C
        bits 0:31
                Decoder horizontal Y alias register 3
        ---------------
        2810
        bits 0:31
                Decoder horizontal Y alias register 4
        ---------------
        2814
```

```
bits 0:31
        Decoder horizontal Y alias register 5
---------------
2818
bits 0:31
        Decoder horizontal Y alias trigger

These six registers control the horizontal aliasing filter for the Y plane.
The first five registers must all be loaded before accessing the trigger
(2818), as this register actually clocks the data through for the first
five.

To correctly program set the filter, this whole procedure must be done 16
times. The actual register contents are copied from a lookup-table in the
firmware which contains 4 different filter settings.

--------------------------------------------------------------------------------
281C
bits 0:31
        Decoder horizontal UV alias register 1
---------------
2820
bits 0:31
        Decoder horizontal UV alias register 2
---------------
2824
bits 0:31
        Decoder horizontal UV alias register 3
---------------
2828
bits 0:31
        Decoder horizontal UV alias register 4
---------------
282C
bits 0:31
        Decoder horizontal UV alias register 5
---------------
2830
bits 0:31
        Decoder horizontal UV alias trigger

These six registers control the horizontal aliasing for the UV plane.
Operation is the same as the Y filter, with 2830 being the trigger
register.

--------------------------------------------------------------------------------
2834
bits 0:15
        Decoder Y source width in pixels

bits 16:31
        Decoder Y destination width in pixels
---------------
2838
bits 0:15
        Decoder UV source width in pixels

bits 16:31
        Decoder UV destination width in pixels

NOTE: For both registers, the resulting image must be fully visible on
screen. If the image exceeds the right edge both the source and destination
size must be adjusted to reflect the visible portion. For the source width,
you must take into account the scaling when calculating the new value.
--------------------------------------------------------------------------------

283C
bits 0:31
        Decoder Y horizontal scaling
                Normally = Reg 2854 >> 2
---------------
2840
bits 0:31
        Decoder ?? unknown - horizontal scaling
        Usually 0x00080514
---------------
2844
bits 0:31
        Decoder UV horizontal scaling
        Normally = Reg 2854 >> 2
---------------
```

```
2848
bits 0:31
        Decoder ?? unknown - horizontal scaling
        Usually 0x00100514
---------------
284C
bits 0:31
        Decoder ?? unknown - Y plane
        Usually 0x00200020
---------------
2850
bits 0:31
        Decoder ?? unknown - UV plane
        Usually 0x00200020
---------------
2854
bits 0:31
        Decoder 'master' value for horizontal scaling
---------------
2858
bits 0:31
        Decoder ?? unknown
        Usually 0
---------------
285C
bits 0:31
        Decoder ?? unknown
        Normally = Reg 2854 >> 1
---------------
2860
bits 0:31
        Decoder ?? unknown
        Usually 0
---------------
2864
bits 0:31
        Decoder ?? unknown
        Normally = Reg 2854 >> 1
---------------
2868
bits 0:31
        Decoder ?? unknown
        Usually 0

Most of these registers either control horizontal scaling, or appear linked
to it in some way. Register 2854 contains the 'master' value & the other
registers can be calculated from that one. You must also remember to
correctly set the divider in Reg 2874.

To enlarge:
        Reg 2854 = (source_width * 0x00200000) / destination_width
        Reg 2874 = No divide

To reduce from full size down to half size:
        Reg 2854 = (source_width/2 * 0x00200000) / destination width
        Reg 2874 = Divide by 2

To reduce from half size down to quarter size:
        Reg 2854 = (source_width/4 * 0x00200000) / destination width
        Reg 2874 = Divide by 4

The result is always rounded up.

--------------------------------------------------------------------------------
286C
bits 0:15
        Decoder horizontal Y buffer offset

bits 15:31
        Decoder horizontal UV buffer offset

Offset into the video image buffer. If the offset is gradually incremented,
the on screen image will move left & wrap around higher up on the right.

--------------------------------------------------------------------------------
2870
bits 0:15
        Decoder horizontal Y output offset

bits 16:31
        Decoder horizontal UV output offset
```

```
Offsets the actual video output. Controls output alignment of the Y & UV
planes. The higher the value, the greater the shift to the left. Use
reg 2890 to move the image right.

---------------------------------------------------------------------------------
2874
bits 0:1
        Decoder horizontal Y output size divider
        00 = No divide
        01 = Divide by 2
        10 = Divide by 3

bits 4:5
        Decoder horizontal UV output size divider
        00 = No divide
        01 = Divide by 2
        10 = Divide by 3

bit 8
        Decoder ?? unknown
        0 = Normal
        1 = Affects video output levels

bit 16
        Decoder ?? unknown
        0 = Normal
        1 = Disable horizontal filter

---------------------------------------------------------------------------------
2878
bit 0
        ?? unknown

bit 1
        osd on/off
        0 = osd off
        1 = osd on

bit 2
        Decoder + osd video timing
        0 = NTSC
        1 = PAL

bits 3:4
        ?? unknown

bit 5
        Decoder + osd
        Swaps upper & lower fields

---------------------------------------------------------------------------------
287C
bits 0:10
        Decoder & osd ?? unknown
        Moves entire screen horizontally. Starts at 0x005 with the screen
        shifted heavily to the right. Incrementing in steps of 0x004 will
        gradually shift the screen to the left.

bits 11:31
        ?? unknown

Normally contents are 0x00101111 (NTSC) or 0x1010111d (PAL)

---------------------------------------------------------------------------------
2880  --------    ?? unknown
2884  --------    ?? unknown
---------------------------------------------------------------------------------
2888
bit 0
        Decoder + osd ?? unknown
        0 = Normal
        1 = Misaligned fields (Correctable through 289C & 28A4)

bit 4
        ?? unknown

bit 8
        ?? unknown

Warning: Bad values will require a firmware reload to recover.
```

```
                Known to be bad are 0x000,0x011,0x100,0x111
--------------------------------------------------------------------------------
288C
bits 0:15
        osd ?? unknown
        Appears to affect the osd position stability. The higher the value the
        more unstable it becomes. Decoder output remains stable.

bits 16:31
        osd ?? unknown
        Same as bits 0:15

--------------------------------------------------------------------------------
2890
bits 0:11
        Decoder output horizontal offset.

Horizontal offset moves the video image right. A small left shift is
possible, but it's better to use reg 2870 for that due to its greater
range.

NOTE: Video corruption will occur if video window is shifted off the right
edge. To avoid this read the notes for 2834 & 2838.
--------------------------------------------------------------------------------
2894
bits 0:23
        Decoder output video surround colour.

Contains the colour (in yuv) used to fill the screen when the video is
running in a window.
--------------------------------------------------------------------------------
2898
bits 0:23
        Decoder video window colour
        Contains the colour (in yuv) used to fill the video window when the
        video is turned off.

bit 24
        Decoder video output
        0 = Video on
        1 = Video off

bit 28
        Decoder plane order
        0 = Y,UV
        1 = UV,Y

bit 29
        Decoder second plane byte order
        0 = Normal (UV)
        1 = Swapped (VU)

In normal usage, the first plane is Y & the second plane is UV. Though the
order of the planes can be swapped, only the byte order of the second plane
can be swapped. This isn't much use for the Y plane, but can be useful for
the UV plane.

--------------------------------------------------------------------------------
289C
bits 0:15
        Decoder vertical field offset 1

bits 16:31
        Decoder vertical field offset 2

Controls field output vertical alignment. The higher the number, the lower
the image on screen. Known starting values are 0x011E0017 (NTSC) &
0x01500017 (PAL)
--------------------------------------------------------------------------------
28A0
bits 0:15
        Decoder & osd width in pixels

bits 16:31
        Decoder & osd height in pixels

All output from the decoder & osd are disabled beyond this area. Decoder
output will simply go black outside of this region. If the osd tries to
exceed this area it will become corrupt.
--------------------------------------------------------------------------------
28A4
```

```
bits 0:11
        osd left shift.

Has a range of 0x770->0x7FF. With the exception of 0, any value outside of
this range corrupts the osd.
--------------------------------------------------------------------------------
28A8
bits 0:15
        osd vertical field offset 1

bits 16:31
        osd vertical field offset 2

Controls field output vertical alignment. The higher the number, the lower
the image on screen. Known starting values are 0x011E0017 (NTSC) &
0x01500017 (PAL)
--------------------------------------------------------------------------------
28AC  --------    ?? unknown
|
V
28BC  --------    ?? unknown
--------------------------------------------------------------------------------
28C0
bit 0
        Current output field
        0 = first field
        1 = second field

bits 16:31
        Current scanline
        The scanline counts from the top line of the first field
        through to the last line of the second field.
--------------------------------------------------------------------------------
28C4  --------    ?? unknown
|
V
28F8  --------    ?? unknown
--------------------------------------------------------------------------------
28FC
bit 0
        ?? unknown
        0 = Normal
        1 = Breaks decoder & osd output
--------------------------------------------------------------------------------
2900
bits 0:31
        Decoder vertical Y alias register 1
---------------
2904
bits 0:31
        Decoder vertical Y alias register 2
---------------
2908
bits 0:31
        Decoder vertical Y alias trigger

These three registers control the vertical aliasing filter for the Y plane.
Operation is similar to the horizontal Y filter (2804). The only real
difference is that there are only two registers to set before accessing
the trigger register (2908). As for the horizontal filter, the values are
taken from a lookup table in the firmware, and the procedure must be
repeated 16 times to fully program the filter.
--------------------------------------------------------------------------------
290C
bits 0:31
        Decoder vertical UV alias register 1
---------------
2910
bits 0:31
        Decoder vertical UV alias register 2
---------------
2914
bits 0:31
        Decoder vertical UV alias trigger

These three registers control the vertical aliasing filter for the UV
plane. Operation is the same as the Y filter, with 2914 being the trigger.
--------------------------------------------------------------------------------
2918
bits 0:15
        Decoder Y source height in pixels
```

```
bits 16:31
        Decoder Y destination height in pixels
---------------
291C
bits 0:15
        Decoder UV source height in pixels divided by 2

bits 16:31
        Decoder UV destination height in pixels

NOTE: For both registers, the resulting image must be fully visible on
screen. If the image exceeds the bottom edge both the source and
destination size must be adjusted to reflect the visible portion. For the
source height, you must take into account the scaling when calculating the
new value.
--------------------------------------------------------------------------------
2920
bits 0:31
        Decoder Y vertical scaling
        Normally = Reg 2930 >> 2
---------------
2924
bits 0:31
        Decoder Y vertical scaling
        Normally = Reg 2920 + 0x514
---------------
2928
bits 0:31
        Decoder UV vertical scaling
        When enlarging = Reg 2930 >> 2
        When reducing = Reg 2930 >> 3
---------------
292C
bits 0:31
        Decoder UV vertical scaling
        Normally = Reg 2928 + 0x514
---------------
2930
bits 0:31
        Decoder 'master' value for vertical scaling
---------------
2934
bits 0:31
        Decoder ?? unknown - Y vertical scaling
---------------
2938
bits 0:31
        Decoder Y vertical scaling
        Normally = Reg 2930
---------------
293C
bits 0:31
        Decoder ?? unknown - Y vertical scaling
---------------
2940
bits 0:31
        Decoder UV vertical scaling
        When enlarging = Reg 2930 >> 1
        When reducing = Reg 2930
---------------
2944
bits 0:31
        Decoder ?? unknown - UV vertical scaling
---------------
2948
bits 0:31
        Decoder UV vertical scaling
        Normally = Reg 2940
---------------
294C
bits 0:31
        Decoder ?? unknown - UV vertical scaling

Most of these registers either control vertical scaling, or appear linked
to it in some way. Register 2930 contains the 'master' value & all other
registers can be calculated from that one. You must also remember to
correctly set the divider in Reg 296C

To enlarge:
        Reg 2930 = (source_height * 0x00200000) / destination_height
        Reg 296C = No divide

To reduce from full size down to half size:
        Reg 2930 = (source_height/2 * 0x00200000) / destination height
        Reg 296C = Divide by 2

To reduce from half down to quarter.
        Reg 2930 = (source_height/4 * 0x00200000) / destination height
        Reg 296C = Divide by 4

--------------------------------------------------------------------------------
2950
bits 0:15
        Decoder Y line index into display buffer, first field

bits 16:31
        Decoder Y vertical line skip, first field
--------------------------------------------------------------------------------
2954
bits 0:15
        Decoder Y line index into display buffer, second field

bits 16:31
        Decoder Y vertical line skip, second field
--------------------------------------------------------------------------------
2958
bits 0:15
        Decoder UV line index into display buffer, first field

bits 16:31
        Decoder UV vertical line skip, first field
--------------------------------------------------------------------------------
295C
bits 0:15
        Decoder UV line index into display buffer, second field

bits 16:31
        Decoder UV vertical line skip, second field
--------------------------------------------------------------------------------
2960
bits 0:15
        Decoder destination height minus 1

bits 16:31
        Decoder destination height divided by 2
--------------------------------------------------------------------------------
2964
bits 0:15
        Decoder Y vertical offset, second field

bits 16:31
        Decoder Y vertical offset, first field

These two registers shift the Y plane up. The higher the number, the
greater the shift.
--------------------------------------------------------------------------------
2968
bits 0:15
        Decoder UV vertical offset, second field

bits 16:31
        Decoder UV vertical offset, first field

These two registers shift the UV plane up. The higher the number, the
greater the shift.
--------------------------------------------------------------------------------
296C
bits 0:1
        Decoder vertical Y output size divider
        00 = No divide
        01 = Divide by 2
        10 = Divide by 4

bits 8:9
        Decoder vertical UV output size divider
        00 = No divide
        01 = Divide by 2
        10 = Divide by 4
--------------------------------------------------------------------------------
2970
bit 0
```

```
        Decoder ?? unknown
        0 = Normal
        1 = Affect video output levels

bit 16
        Decoder ?? unknown
        0 = Normal
        1 = Disable vertical filter

--------------------------------------------------------------------------------
2974  --------   ?? unknown
|
V
29EF  --------   ?? unknown
--------------------------------------------------------------------------------
2A00
bits 0:2
        osd colour mode
        000 = 8 bit indexed
        001 = 16 bit (565)
        010 = 15 bit (555)
        011 = 12 bit (444)
        100 = 32 bit (8888)

bits 4:5
        osd display bpp
        01 = 8 bit
        10 = 16 bit
        11 = 32 bit

bit 8
        osd global alpha
        0 = Off
        1 = On

bit 9
        osd local alpha
        0 = Off
        1 = On

bit 10
        osd colour key
        0 = Off
        1 = On

bit 11
        osd ?? unknown
        Must be 1

bit 13
        osd colour space
        0 = ARGB
        1 = AYVU

bits 16:31
        osd ?? unknown
        Must be 0x001B (some kind of buffer pointer ?)

When the bits-per-pixel is set to 8, the colour mode is ignored and
assumed to be 8 bit indexed. For 16 & 32 bits-per-pixel the colour depth
is honoured, and when using a colour depth that requires fewer bytes than
allocated the extra bytes are used as padding. So for a 32 bpp with 8 bit
index colour, there are 3 padding bytes per pixel. It's also possible to
select 16bpp with a 32 bit colour mode. This results in the pixel width
being doubled, but the color key will not work as expected in this mode.

Colour key is as it suggests. You designate a colour which will become
completely transparent. When using 565, 555 or 444 colour modes, the
colour key is always 16 bits wide. The colour to key on is set in Reg 2A18.

Local alpha works differently depending on the colour mode. For 32bpp & 8
bit indexed, local alpha is a per-pixel 256 step transparency, with 0 being
transparent and 255 being solid. For the 16bpp modes 555 & 444, the unused
bit(s) act as a simple transparency switch, with 0 being solid & 1 being
fully transparent. There is no local alpha support for 16bit 565.

Global alpha is a 256 step transparency that applies to the entire osd,
with 0 being transparent & 255 being solid.

It's possible to combine colour key, local alpha & global alpha.
--------------------------------------------------------------------------------
```

```
2A04
bits 0:15
        osd x coord for left edge

bits 16:31
        osd y coord for top edge
---------------
2A08
bits 0:15
        osd x coord for right edge

bits 16:31
        osd y coord for bottom edge

For both registers, (0,0) = top left corner of the display area. These
registers do not control the osd size, only where it's positioned & how
much is visible. The visible osd area cannot exceed the right edge of the
display, otherwise the osd will become corrupt. See reg 2A10 for
setting osd width.
--------------------------------------------------------------------------------
2A0C
bits 0:31
        osd buffer index

An index into the osd buffer. Slowly incrementing this moves the osd left,
wrapping around onto the right edge
--------------------------------------------------------------------------------
2A10
bits 0:11
        osd buffer 32 bit word width

Contains the width of the osd measured in 32 bit words. This means that all
colour modes are restricted to a byte width which is divisible by 4.
--------------------------------------------------------------------------------
2A14
bits 0:15
        osd height in pixels

bits 16:32
        osd line index into buffer
        osd will start displaying from this line.
--------------------------------------------------------------------------------
2A18
bits 0:31
        osd colour key

Contains the colour value which will be transparent.
--------------------------------------------------------------------------------
2A1C
bits 0:7
        osd global alpha

Contains the global alpha value (equiv ivtvfbctl --alpha XX)
--------------------------------------------------------------------------------
2A20  --------    ?? unknown
|
V
2A2C  --------    ?? unknown
--------------------------------------------------------------------------------
2A30
bits 0:7
        osd colour to change in indexed palette
---------------
2A34
bits 0:31
        osd colour for indexed palette

To set the new palette, first load the index of the colour to change into
2A30, then load the new colour into 2A34. The full palette is 256 colours,
so the index range is 0x00-0xFF
--------------------------------------------------------------------------------
2A38  --------    ?? unknown
2A3C  --------    ?? unknown
--------------------------------------------------------------------------------
2A40
bits 0:31
        osd ?? unknown

Affects overall brightness, wrapping around to black
--------------------------------------------------------------------------------
2A44
```

```
bits 0:31
        osd ?? unknown

Green tint
--------------------------------------------------------------------------------
2A48
bits 0:31
        osd ?? unknown

Red tint
--------------------------------------------------------------------------------
2A4C
bits 0:31
        osd ?? unknown

Affects overall brightness, wrapping around to black
--------------------------------------------------------------------------------
2A50
bits 0:31
        osd ?? unknown

Colour shift
--------------------------------------------------------------------------------
2A54
bits 0:31
        osd ?? unknown

Colour shift
--------------------------------------------------------------------------------
2A58  --------    ?? unknown
|
V
2AFC  --------    ?? unknown
--------------------------------------------------------------------------------
2B00
bit 0
        osd filter control
        0 = filter off
        1 = filter on

bits 1:4
        osd ?? unknown

--------------------------------------------------------------------------------
```

The cx231xx DMA engine

This page describes the structures and procedures used by the cx2341x DMA engine.

Introduction

The cx2341x PCI interface is busmaster capable. This means it has a DMA engine to efficiently transfer large volumes of data between the card and main memory without requiring help from a CPU. Like most hardware, it must operate on contiguous physical memory. This is difficult to come by in large quantities on virtual memory machines.

Therefore, it also supports a technique called "scatter-gather". The card can transfer multiple buffers in one operation. Instead of allocating one large contiguous buffer, the driver can allocate several smaller buffers.

In practice, I've seen the average transfer to be roughly 80K, but transfers above 128K were not uncommon, particularly at startup. The 128K figure is important, because that is the largest block that the kernel can normally allocate. Even still, 128K blocks are hard to come by, so the driver writer is urged to choose a smaller block size and learn the scatter-gather technique.

Mailbox #10 is reserved for DMA transfer information.

Note: the hardware expects little-endian data ('intel format').

Flow

This section describes, in general, the order of events when handling DMA transfers. Detailed information follows this section.

- The card raises the Encoder interrupt.
- The driver reads the transfer type, offset and size from Mailbox #10.
- The driver constructs the scatter-gather array from enough free dma buffers to cover the size.
- The driver schedules the DMA transfer via the ScheduleDMAtoHost API call.
- The card raises the DMA Complete interrupt.
- The driver checks the DMA status register for any errors.

- The driver post-processes the newly transferred buffers.

NOTE! It is possible that the Encoder and DMA Complete interrupts get raised simultaneously. (End of the last, start of the next, etc.)

Mailbox #10

The Flags, Command, Return Value and Timeout fields are ignored.

- Name: Mailbox #10
- Results[0]: Type: 0: MPEG.
- Results[1]: Offset: The position relative to the card's memory space.
- Results[2]: Size: The exact number of bytes to transfer.

My speculation is that since the StartCapture API has a capture type of "RAW" available, that the type field will have other values that correspond to YUV and PCM data.

Scatter-Gather Array

The scatter-gather array is a contiguously allocated block of memory that tells the card the source and destination of each data-block to transfer. Card "addresses" are derived from the offset supplied by Mailbox #10. Host addresses are the physical memory location of the target DMA buffer.

Each S-G array element is a struct of three 32-bit words. The first word is the source address, the second is the destination address. Both take up the entire 32 bits. The lowest 18 bits of the third word is the transfer byte count. The high-bit of the third word is the "last" flag. The last-flag tells the card to raise the DMA_DONE interrupt. From hard personal experience, if you forget to set this bit, the card will still "work" but the stream will most likely get corrupted.

The transfer count must be a multiple of 256. Therefore, the driver will need to track how much data in the target buffer is valid and deal with it accordingly.

Array Element:

- 32-bit Source Address
- 32-bit Destination Address
- 14-bit reserved (high bit is the last flag)
- 18-bit byte count

DMA Transfer Status

Register 0x0004 holds the DMA Transfer Status:

- bit 0: read completed
- bit 1: write completed
- bit 2: DMA read error
- bit 3: DMA write error
- bit 4: Scatter-Gather array error