This document 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:

```
        ivtvctl -0 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.

```
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.

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

Missing
Encoder API call completed
Decoder API call completed
Encoder API post(?)
Decoder API post(?)
Decoder VTRACE event