```
struct dma_chan         chan;/**
 * struct mv61_pdma_chan - physical channel control
 * @ch_regs: virtual base address of channel's mv61_pdma_chan_regs
 * @vtype: channel type, same as id/type of virtual controllers
 * @index: this physical channel's index
 * @mv61p: top level control structure
 */
struct mv61_pdma_chan{
        void __iomem            *ch_regs;
        enum mv61_vdma_type     vtype;
int                     index;
        struct mv61_dma         *mv61p;

        /* these elements must be protected by struct mv61_dma.biglock */
};
```

struct mv61_pdma_chan model cdma中的physical channel.比如在99PA6270中一个cdma controller有8个channel(0 - 7).

比较简单，因为与dmaengine framework打交道的是struct mv61_vdma_chan (virtual channel)。

struct mv61_pdma_chan实体是被包含在struct mv61_dma中的。

```
struct mv61_dma {
        struct device           *dev;
        void __iomem            *ch_regs[MV61_DMA_MAX_NR_PCHANNELS];
        void __iomem            *CDMAInt;
        int                     pchannels;
        u32                     irq_call_cnt;
        struct tasklet_struct   tasklet;

        struct mv61_dma_dispatch *dispatch;
        struct mv61_dma_vpmap   *vpmap;
        struct kmem_cache       *desc_cachep;
        struct kmem_cache       *chain_cachep;
        struct mv61_vdma        *mv61v[MV61_NR_VDMA_CONTROLLERS];

        spinlock_t              all_chains_lock;
        struct list_head        all_chains;

        spinlock_t              biglock;
        int reva;
struct mv61_pdma_chan   chan[0];
};
```

这里的index就是chan[index] ==> physical dma channel

void __iomem *ch_regs;

该channel的physical address

enum mv61_vdma_type vtype;

在cdma driver中，virtual channel被分成了如下4中类型

```
/**
 * enum mv61_vdma_type - index for virtual dma controllers
 * @MV61_VDMA_OWNED:  Each virtual channel owns a physical channel
 * @MV61_VDMA_SHARED: Virtual channels share a pool of physical channels
 * @MV61_VDMA_MEMOPS: Virtual channel(s) for kernel use
 */
enum mv61_vdma_type {
        MV61_VDMA_OWNED = 0,
        MV61_VDMA_SHARED,
        MV61_VDMA_CYCLIC,
        MV61_VDMA_MEMOPS,
        MV61_VDMA_UNASSIGNED,
};
```

这里的vtype表示与该physical channel对应的virtual channel的类型。毕竟一个virtual channel只有真正对应到某个physical channel时才能工作，否则不就是空中楼阁吗？

int index;

在88PA6270中就是0-7的index

struct mv61_dma *mv61p;

该physical channel所属的cdma device。

```
/**
 * struct mv61_vdma_chan - virtual channel control
 * @chan: channel control structure used by dmaengine API
 * @mv61v: virtual controller instance that owns this channel
 * @def: default reg subset settings to load into a physical channel
 * @wrap: word aligned byte count for dest address to wrap
 * @lock: protect everything except def and chan (which has its own lock)
 * @irqs: TBD
 * @completed: TBD
 * @active_list: TBD
 * @complete_list: completed subtransactions
 * @queue: TBD
 * @hwstat:
 * @vcstatus:
 *
 * Channel index into mv61_dma_vpmap.v_to_p will be offset for each instance.
 * This channel index within its instance is contained in chan.chan_id.
 */
struct mv61_vdma_chan{
        struct dma_chan         chan;
        struct mv61_vdma        *mv61v;

        struct mv61_vreg_defs   def;
        int                     wrap;

        spinlock_t              lock;

        /* these elements are all protected by lock */
        u32                     irqs;
        dma_cookie_t            completed;
        dma_cookie_t            started;
        struct list_head        active_list;
        struct list_head        complete_list;
        struct list_head        queue;
        struct mv61_stat_regs   hwstat;
        int                     residue;
        enum dma_status         status;

};
```

struct mv61_vdma_chan model virtual dma channel.

virtual channel可以远远多于实际存在的physical channel。同时与与dmaengine framework中对应的是

virtual channel，而不是physical channel。所以mv61 cdma driver中与damengine framework交互的

是virtual channel。所以整个driver除了驱动cdma physical channel的code，还有很大一部分是virtual

channel与physical channel之间的管理。

struct dma_chan chan;

dmaengine framework中的channel.

struct mv61_vdma *mv61v;

该virtual channel所属的cdma device。

struct mv61_vreg_defs def;

这是针对该virtual channel的setting,当然最终还是要设置到0-7的某个physical channel中去的。

struct mv61_stat_regs hwstat;

当该virtual channel与physical channel绑定并工作后的status info，实际上就是绑定的physical channel的

status info。

enum dma_status status;

```
/**
 * enum dma_status - DMA transaction status
 * @DMA_COMPLETE: transaction completed
 * @DMA_IN_PROGRESS: transaction not yet processed
 * @DMA_PAUSED: transaction is paused
 * @DMA_ERROR: transaction failed
 */
enum dma_status {
```

```
    DMA_COMPLETE,
    DMA_IN_PROGRESS,
    DMA_PAUSED,
    DMA_ERROR,
};
```

```
    struct list_head    active_list;
    struct list_head    complete_list;
    struct list_head    queue;
```

这3个list上挂的是struct mv61_desc node。

queue list上挂的是提交的struct mv61_desc node.

```
/*
 * mv61vc_tx_submit - dmaengine API to submit prepared transactions to queue.
 * @tx: dmaengine API subset of transaction descriptor
 */
dma_cookie_t mv61vc_tx_submit(struct dma_async_tx_descriptor *tx)
{
struct mv61_desc                *desc = txd_to_mv61_desc(tx);        ①
        struct mv61_vdma_chan           *mv61vc = dchan_to_mv61_vdma_chan(tx->cha
n);
        dma_cookie_t                    cookie;
        unsigned long                   lockvcflags;

        spin_lock_irqsave(&mv61vc->lock, lockvcflags);
        cookie = mv61vc_assign_cookie(mv61vc, desc);
        list_add_tail(&desc->desc_node, &mv61vc->queue);    `②

        __dev_vdbg(chan2dev(tx->chan), "tx_submit: queued desc %p cookie %u\n",
                 desc, desc->txd.cookie);
        if(vdumptx && desc)
                mv61_tx_dump(desc);
        spin_unlock_irqrestore(&mv61vc->lock, lockvcflags);

        return cookie;
}
```

①

由damengine中的transaction node得到cdma的descriptot node

②

挂到queue上

complete_list应该是已经完成传输的node。

```
/**
 * struct mv61_dma - top level control structure
 * @device: physical platform device's "device" member
 * @ch_regs[]: base addresses of each channel's register bank
 * @CDMAInt: base address of top interrupt status register
 * @pchannels: total number of physical channels available
 * @irq_call_cnt: cumulative count of irq handler calls for debug
 * @tasklet: bottom half tasklet invoked by interrupt handler
 * @dispatch: pool manager dispatch data
 * @desc_cachep: slab cache for the transaction descriptors
 * @lli_cachep: slab cache for the dma linked list descriptors
 * @mv61v: control structure for each of the virtual dma devices
 * @all_chains_lock: protect all_chains list using spin_lock_bh
 * @all_chains: used for cleanup to avoid mem leaks
 * @biglock: protect physical channels, mapping, dispatch
 * @chan[]: physical channel control structures
 *
 * The all_chains list head is in top struct mv61_dma, but it is never touched
 * in the irq handler, only the tasklet. Really just need atomic list ops.
 * The kernel does provide smp-safe versions with internal spin_locks, but they
 * still aren't bottom-half-safe, so just handle it here.
 */
struct mv61_dma{
        struct device           *dev;
        void __iomem            *ch_regs[MV61_DMA_MAX_NR_PCHANNELS];
        void __iomem            *CDMAInt;
        int                     pchannels;
        u32                     irq_call_cnt;
        struct tasklet_struct   tasklet;

        struct mv61_dma_dispatch *dispatch;
        struct mv61_dma_vpmap   *vpmap;
        struct kmem_cache       *desc_cachep;
        struct kmem_cache       *chain_cachep;
        struct mv61_vdma        *mv61v[MV61_NR_VDMA_CONTROLLERS];

        spinlock_t              all_chains_lock;
        struct list_head        all_chains;

        spinlock_t              biglock;
        int reva;
        struct mv61_pdma_chan   chan[0];
};
```

struct mv61_dma model real cdma controller

void __iomem *ch_regs[MV61_DMA_MAX_NR_PCHANNELS];

88PA6270 SoC中的cdma controller有8个channel,每个channel都有独立的physical address。

void __iomem *CDMAInt;

这是cdma controller的top registers

int pchannels;

该cdma controller有多少个channel，88PA6270为8

struct mv61_vdma *mv61v[MV61_NR_VDMA_CONTROLLERS];

struct mv61_vdma同来model virtual dma controller,而virtual dma controller有分为如下类型

```
/**
 * enum mv61_vdma_type - index for virtual dma controllers
 * @MV61_VDMA_OWNED:  Each virtual channel owns a physical channel
 * @MV61_VDMA_SHARED: Virtual channels share a pool of physical channels
 * @MV61_VDMA_MEMOPS: Virtual channel(s) for kernel use
 */
enum mv61_vdma_type {
        MV61_VDMA_OWNED = 0,
        MV61_VDMA_SHARED,
        MV61_VDMA_CYCLIC,
        MV61_VDMA_MEMOPS,
        MV61_VDMA_UNASSIGNED,
};

#define MV61_NR_VDMA_CONTROLLERS (MV61_VDMA_UNASSIGNED)
```

这里在physical dma controller包含virtual dam controller的信息.

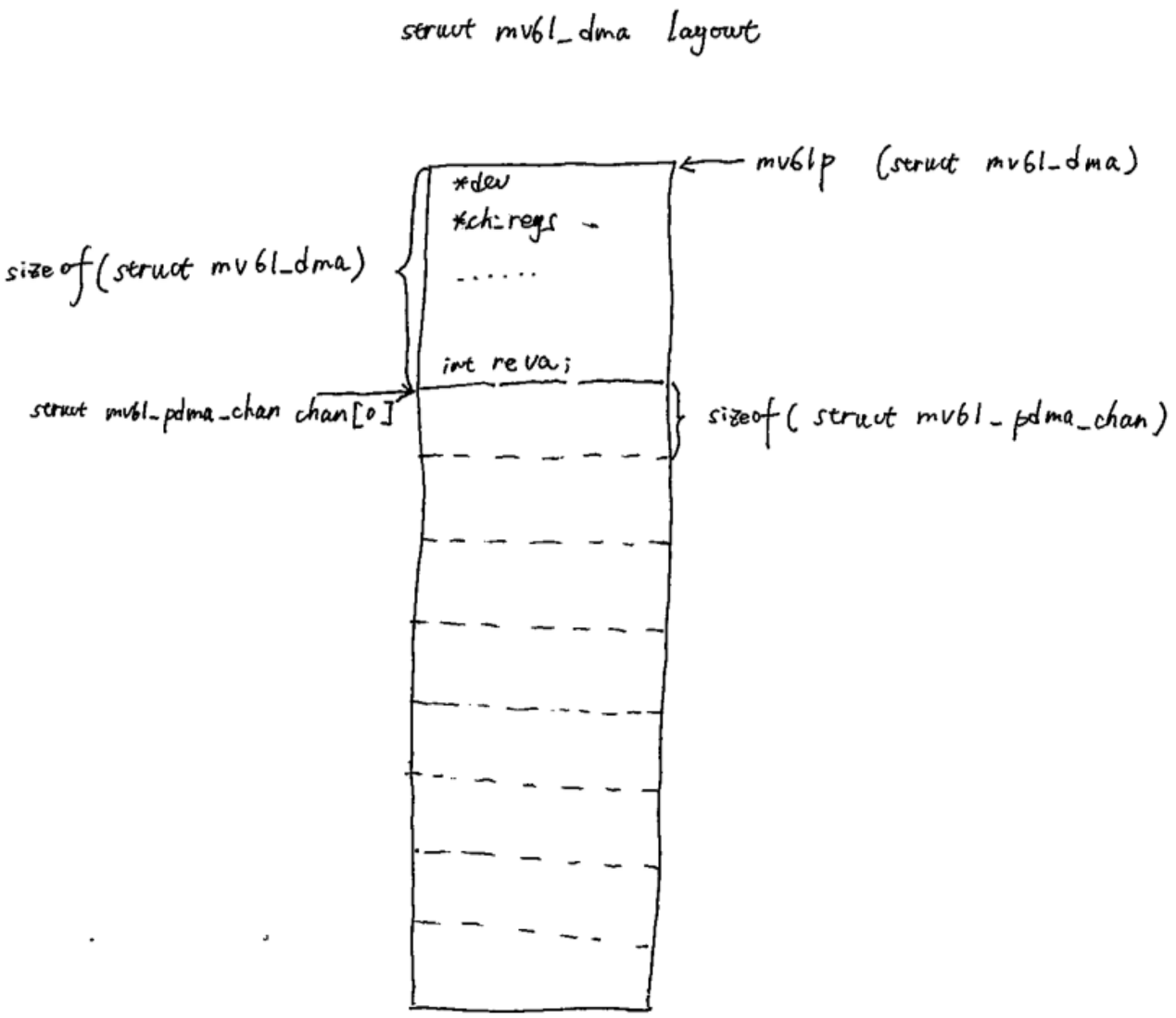
见《dts setting for cdma driver》note。

struct mv61_pdma_chan chan[0];

该cdma controller包含的physical channel。

in mv61_dma_probe()

```
1.          size = sizeof(struct mv61_dma);
2.          size += pdata->nr_channels * sizeof(struct mv61_pdma_chan);
3.
4.          mv61p = devm_kzalloc(&pdev->dev, size, GFP_KERNEL);
```



struct kmem_cache *desc_cachep;

用于allocate struct mv61_desc

struct kmem_cache *chain_cachep;

用于分配struct mv61_chain

struct mv61_dma_vpmap *vpmap;

virtual channel与physical channel之间mapping关系记录

```
/**
 * struct mv61_dma_vpmap - map of virtual <-> physical channel assignment
 * @p_to_v: table of virtual channels assigned to physical channels
 * @voffset: index offset of virtual controller in v_to_p table
 * @v_to_p: table of physical channels assigned to virtual channels
 *
 * TBD whether to lock here or at a higher level.
 */
struct mv61_dma_vpmap{
        /* these elements must be protected by struct mv61_dma.biglock */
        struct mv61_vdma_chan   *p_to_v[MV61_DMA_MAX_NR_PCHANNELS];
        int                     voffset[MV61_NR_VDMA_CONTROLLERS];
        struct mv61_pdma_chan   *v_to_p[0];
};
```

用于建立virtal channel与physical channel之间的mapping.

```
#define MV61_DMA_MAX_NR_PCHANNELS       (12)
#define MV61_DMA_MAX_NR_VCHANNELS       (96)
```

```
struct mv61_vdma_chan   *p_to_v[MV61_DMA_MAX_NR_PCHANNELS];
```

即

struct mv61_vdma_chan *p_to_v[12];

physical channel to virtual channel mapping entries are 12.

physical channel最多是12项。在88PA6270 SoC上只有8 entries。

struct mv61_pdma_chan *v_to_p[0];

具体多少不定，由dts中设定。

in mv61_dma_probe()

```
        /*
         * Create the virtual to physical channel map.
         */
        size = sizeof(struct mv61_dma_vpmap);
        for(i = 0; i < MV61_NR_VDMA_CONTROLLERS; i++)
                size += pdata->nr_virt_chans[i] * sizeof(int);

        mv61p->vpmap = devm_kzalloc(&pdev->dev, size, GFP_KERNEL);
```

这里的

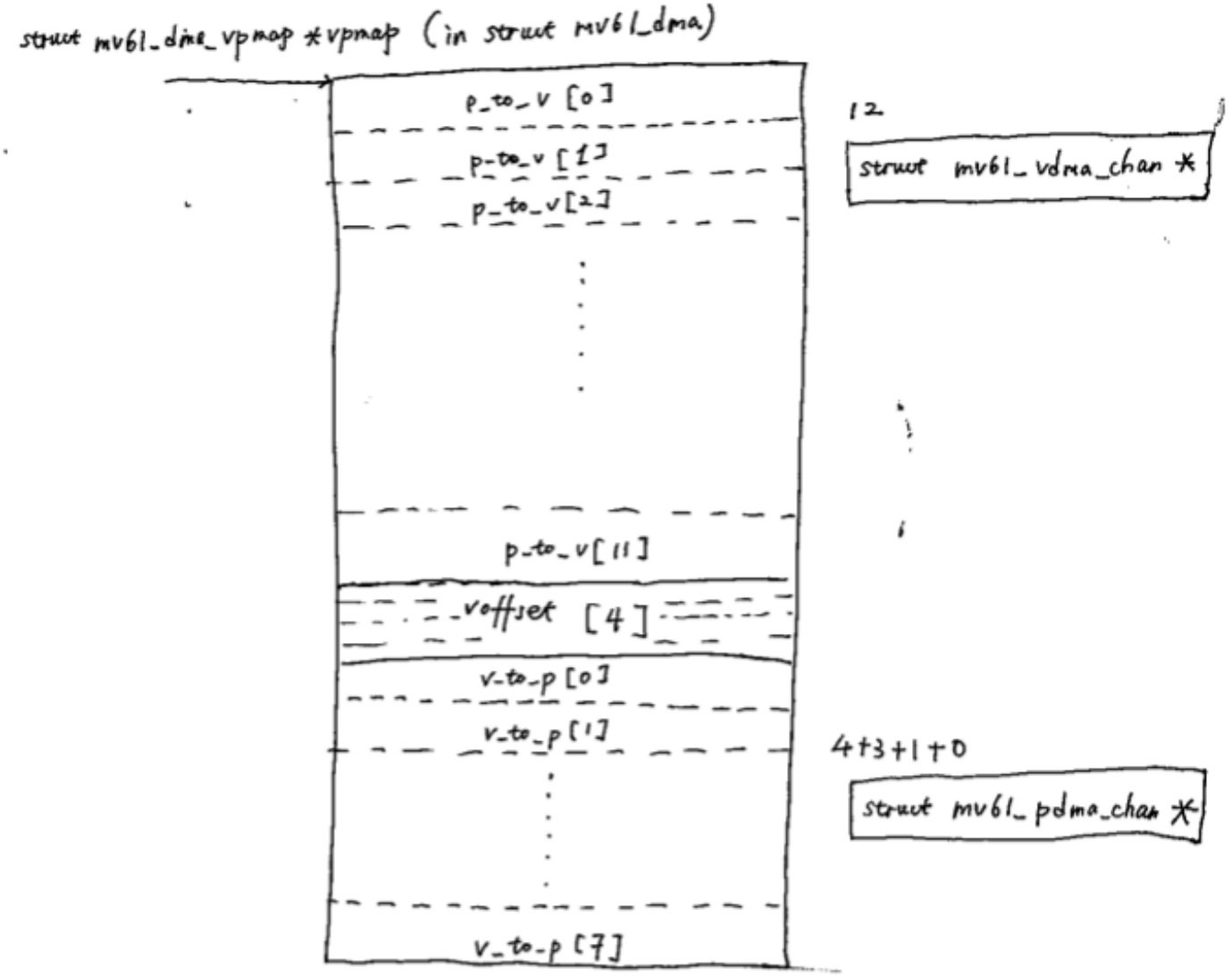
pdata->nr_virt_chans[0]= 4

pdata->nr_virt_chans[1] = 3

pdata->nr_virt_chans[2] = 1

pdata->nr_virt_chans[3] = 0

```
        cdma {
                compatible = "mrvl,mv61_cdma";
                max_owned = <0x4>;
                max_shared = <0x3>;
                max_cyclic = <0x1>;
                max_memops = <0x0>;
                reg = <0x0 0xf9060000 0x0 0x9000>;
                interrupts = <0x0 0xbd 0x4>;
                clocks = <0x32>;
        };
```



具体完成mapping的algorithm在mv61_vpmap_dispatch_init()。