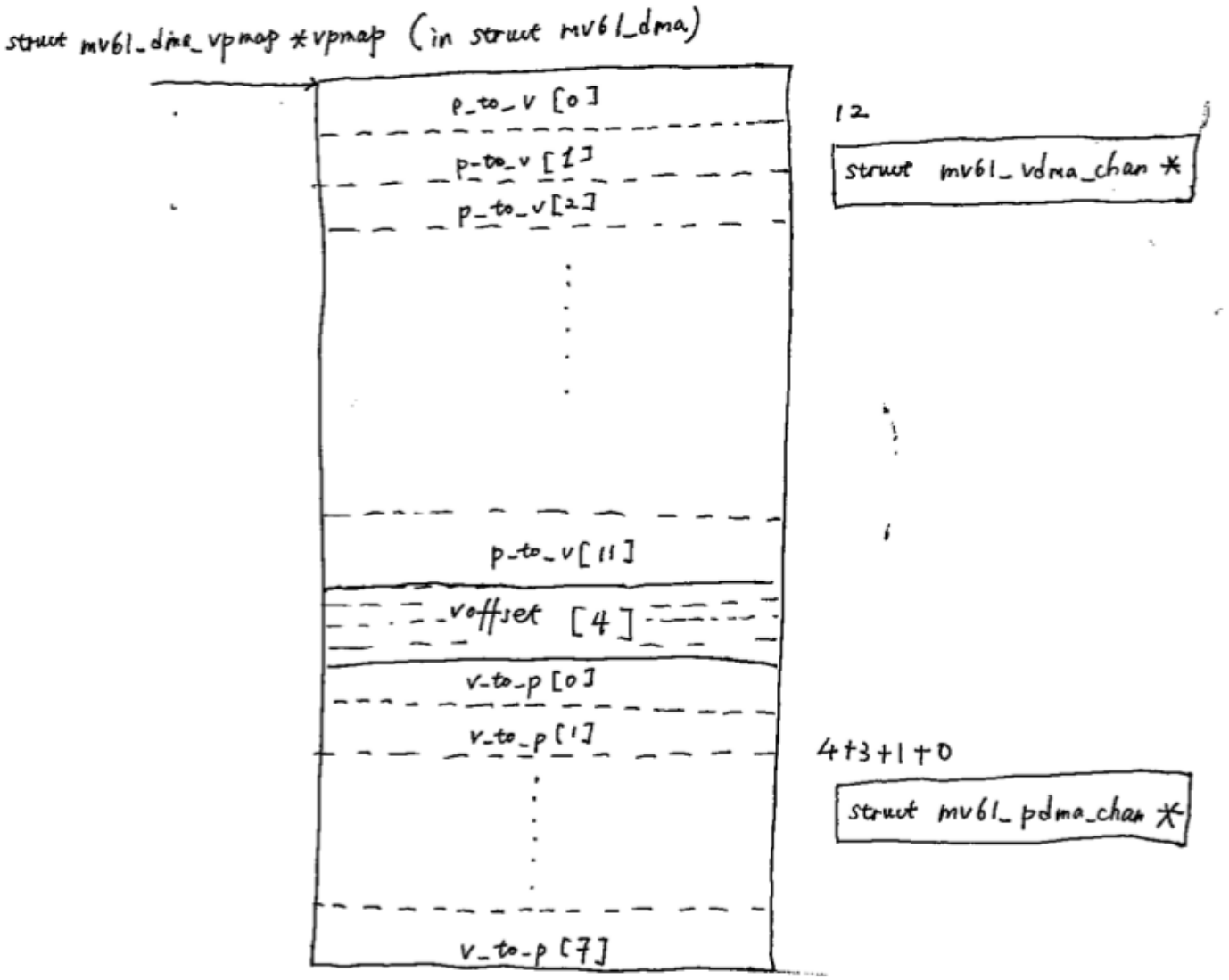


mv61_vpmap_dispatch_init()对上图data structure的initialization.

```
1.  static void __init mv61_vpmap_dispatch_init (struct mv61_vdma *mv61v,
2.                                              struct mv61_dma_platform_data *pdata)
```

mv61v指向当前初始化的virtual dma controller,其中mv61v->vtype指明which controller。

由于当前有4个 virtual dma controller，所以mv61_vpmap_dispatch_init()会被调用4次。

struct mv61_dma_platform_data *pdata

记录了dts中的4个virtual dma controller中channel的分配状况。

```
        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>;
        };
```

pdata->nr_channels = 8

pdata->nr_pool_chans[0] = 4

pdata->nr_pool_chans[1] = 3

pdata->nr_pool_chans[2] = 1

pdata->nr_pool_chans[3] = 0

pdata->nr_virt_chans[0] = 4

pdata->nr_virt_chans[1] = 3

pdata->nr_virt_chans[2] = 1

pdata->nr_virt_chans[3] = 0

```
/**
 * mv61_vpmap_dispatch_init - initialize the channel mapping and dispatcher
 * @mv61v: this top virtual dma control instance
 * @pdata: platform data of physical device
 *
 * Still single-threaded when this is called, but lock to be consistent.
 *
 * Physical channels are assigned in contiguous blocks.
 * All virtual channels that are not SHARED that are directly mapped to
 * corresponding physical channels.
 */
static void __init mv61_vpmap_dispatch_init (struct mv61_vdma *mv61v,
                                             struct mv61_dma_platform_data *pdata)
{
        struct mv61_dma         *mv61p = mv61v->mv61p; /* top phys dma ctrl */
        struct mv61_pdma_chan   *mv61pc;
        struct mv61_vdma_chan   *mv61vc = NULL;
        struct mv61_dma_vpmap   *vpmap = mv61p->vpmap;
        int                     vid = mv61v->vtype;
        int                     i;
        /* offsets within tables for this instance */
        int                     firstp = 0;
        int                     firstv = 0;
        /* absolute index into tables */
        int                     vindex;
        int                     pindex;
        unsigned long           biglockflags;

        spin_lock_irqsave(&mv61p->biglock, biglockflags);

        for (i = 0; i < vid; i++) {                                    ①
```

```
32.                 firstp += pdata->nr_pool_chans[i];
33.                 firstv += pdata->nr_virt_chans[i];
34.         }
35.
36.         vpmap->voffset[vid] = firstv;                    ②
37.
38.         for(i = 0; i < pdata->nr_virt_chans[vid]; i++) {        ③
39.                 vindex = i + firstv;
40.                 vpmap->v_to_p[vindex] = NULL;
41.         }
42.
43.         for (i = 0; i < pdata->nr_pool_chans[vid]; i++) {    ④
44.                 vindex = i + firstv;
45.                 pindex = i + firstp;
46.
47.                 mv61pc = &mv61p->chan[pindex];
48.                 mv61pc->vtype = vid;
49.
50.                 if(vid == MV61_VDMA_SHARED) {
51.                         /* mark the physical channel as available */
52.                         mv61_dispatch_free_pchan(mv61p, mv61pc->index);
53.                         vpmap->p_to_v[pindex] = NULL;
54.                 }
55.                 else {
56.                         /* cross link the physical and virtual channels */
57.                         mv61vc = &mv61v->chan[i];
58.
59.                         vpmap->v_to_p[vindex] = mv61pc;
60.                         vpmap->p_to_v[pindex] = mv61vc;
61.                 }
62.         }
63.
```

```
64.             spin_unlock_irqrestore(&mv61p->biglock, biglockflags);
65.     }
```

①

对MV61_VDMA_OWNED (0)而言，不会进入loop

firstp = 0

firstv = 0

	MV61_VDMA_OWNED(0)	MV61_VDMA_SHARED(1)	MV61_VDMA_CYCLIC(2)	MV61_VDMA_MEMOPS(3)
firstp	0	4	7	8
firstv	0	4	7	8

②

vpmap->voffset[0] = 0

vpmap->voffset[1] = 4

vpmap->voffset[2] = 7

vpmap->voffset[3] = 8

③

当MV61_VDMA_OWNED(0)

vpmap->v_to_p[0] = NULL

vpmap->v_to_p[1] = NULL

vpmap->v_to_p[2] = NULL

vpmap->v_to_p[3] = NULL

当MV61_VDMA_SHARED(1)

vpmap->v_to_p[4] = NULL

vpmap->v_to_p[5] = NULL

vpmap->v_to_p[6] = NULL

当MV61_VDMA_CYCLIC(2)

vpmap->v_to_p[7] = NULL

当MV61_VDMA_MEMOPS(3)

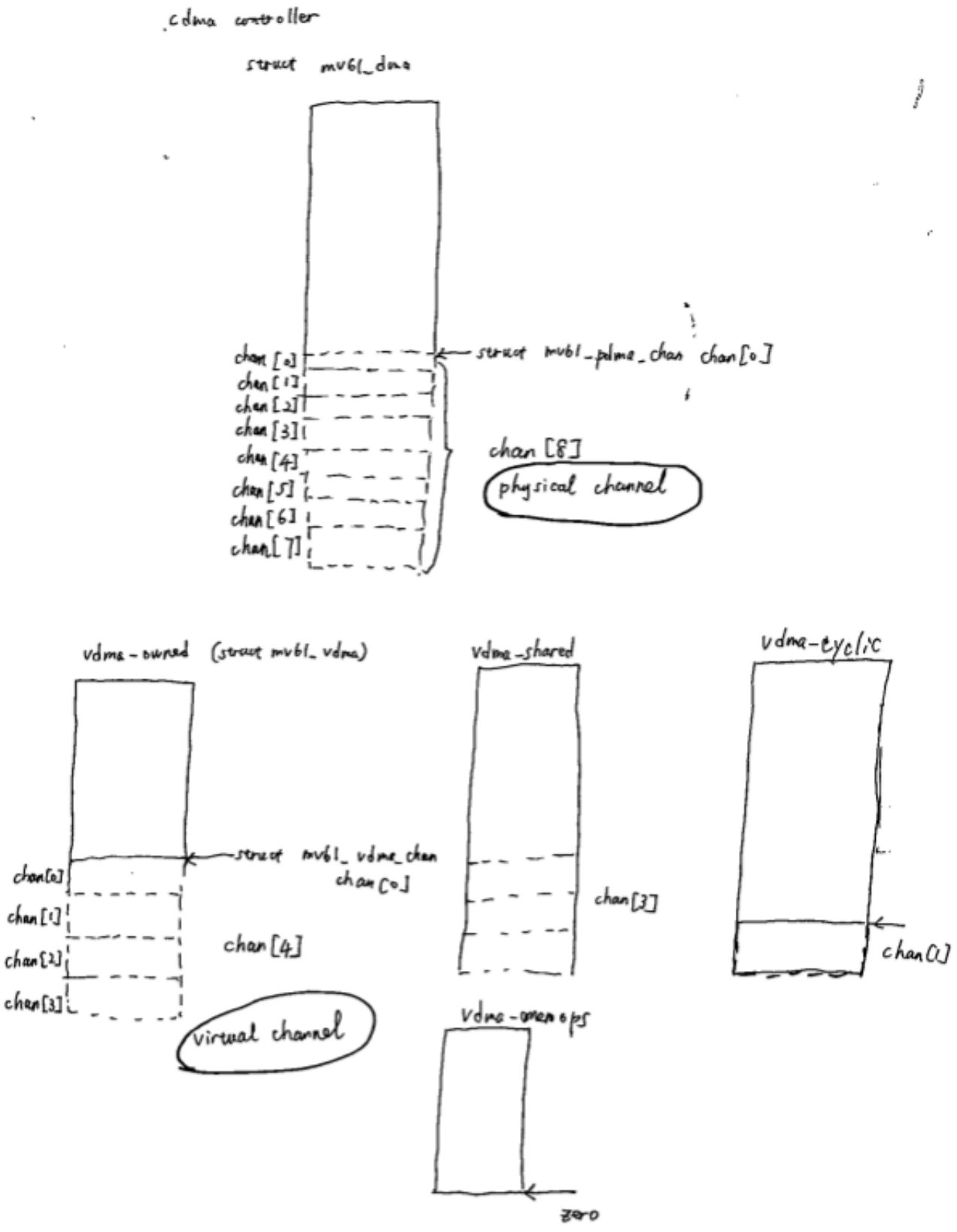
不会进入loop

④

在未建立mapping前virtual channel与physical channel如下

virtual channel v.s. physical channel

两者之间要建立 mapping.



当MV61_VDMA_OWNED(0)

vpmap->v_to_p[0] = mv61p->chan + 0

vpmap->p_to_v[0] = vdma-owned->chan + 0

vpmap->v_to_p[1] = mv61p->chan + 1

vpmap->p_to_v[1] = vdma-owned->chan + 1

vpmap->v_to_p[2] = mv61p->chan + 2

vpmap->p_to_v[2] = vdma-owned->chan + 2

vpmap->v_to_p[3] = mv61p->chan + 3

vpmap->p_to_v[3] = vdma-owned->chan + 3

当MV61_VDMA_SHARED(1)

有点特殊. 其它type的virtual channel 与 physical channel之间是一一对应关系，而shared virtual channel

则是多对一关系。只有shared virtua channel才有dispatch的概念。

由于physical channel是shared，所以不能由physical channel index得到对应的virtual channel

因为是一对多的关系。

而对应的3个physical channel则可被share

```
1.                          /* mark the physical channel as available */
2.                          mv61_dispatch_free_pchan(mv61p, mv61pc->index);
```

这里mv61p代表cdma controller

mv61pc代表physical channel，这里分别为

mv61p->chan[4]

mv61p->chan[5]

mv61p->chan[6]

```
1.  static void mv61_dispatch_free_pchan(struct mv61_dma *mv61p, int pch_index)
2.  {
3.          mv61p->dispatch->pchans |= (1 << pch_index);
4.  }
```

vpmap->p_to_v[4] = NULL

vpmap->p_to_v[5] = NULL

vpmap->p_to_v[6] = NULL

当MV61_VDMA_CYCLIC(2)

vpmap->v_to_p[7] = mv61p->chan + 7

vpmap->p_to_v[7] = vdma-owned->chan + 7

当MV61_VDMA_MEMOPS(3)

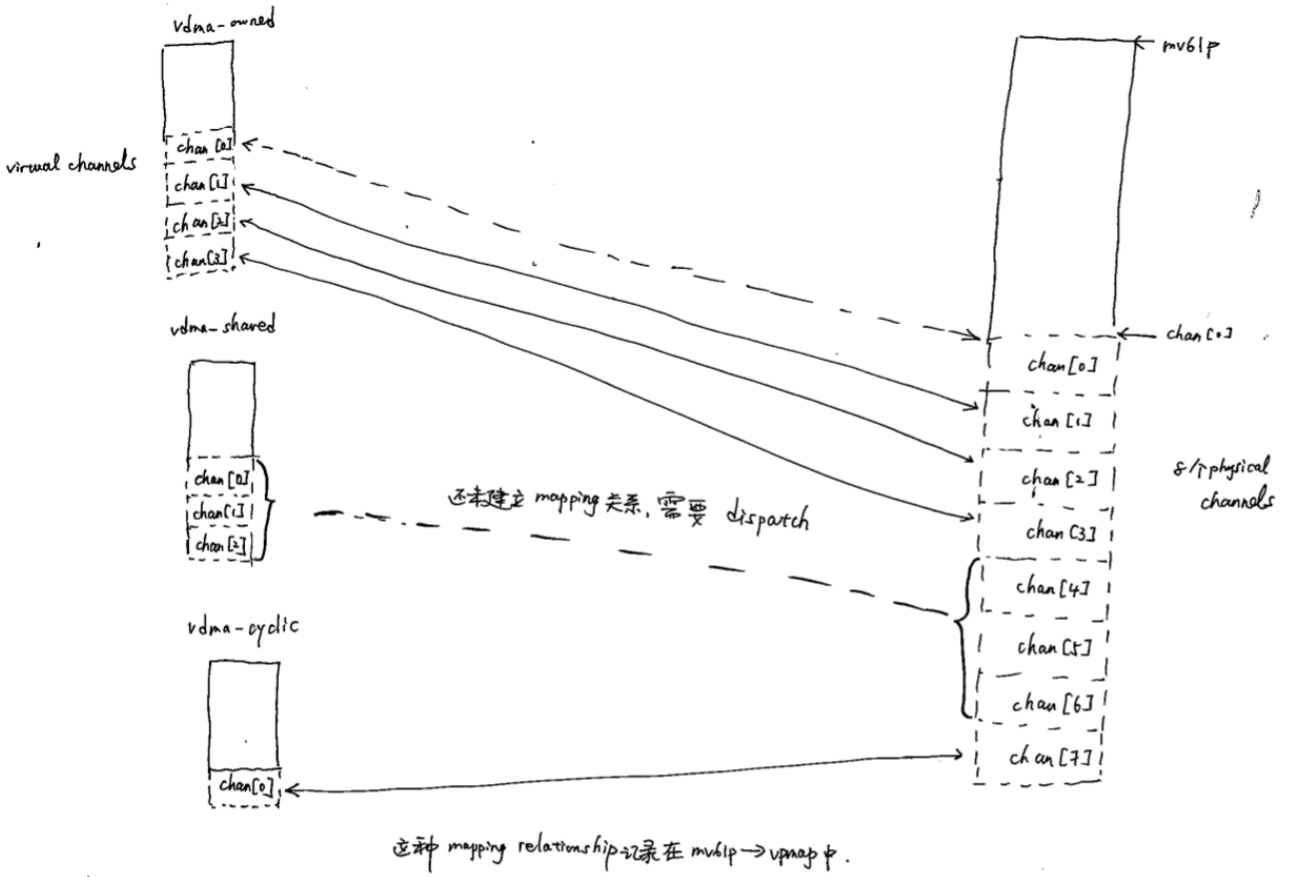
不会进入loop

vpmap->v_to_p[]

由virtual channel index得到physical channel

vpmap->p_to_v[]

由physical channel index得到virtual channel


vdma-owned
virtual channels
chan[0]
chan[1]
chan[2]
chan[3]
mv61p
chan[0]
chan[0]
chan[1]
chan[2]
chan[3]
chan[4]
chan[5]
chan[6]
chan[7]
8个physical channels
vdma-shared
chan[0]
chan[1]
chan[2]
还未建立 mapping 关系，需要 dispatch
vdma-cyclic
chan[0]
这种 mapping relationship 记录在 mv61p → vpmap 中.