```
        int (*device_alloc_chan_resources)(struct dma_chan *chan);
        void (*device_free_chan_resources)(struct dma_chan *chan);

        struct dma_async_tx_descriptor *(*device_prep_dma_memcpy)(
                struct dma_chan *chan, dma_addr_t dest, dma_addr_t src,
                size_t len, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_xor)(
                struct dma_chan *chan, dma_addr_t dest, dma_addr_t *src,
                unsigned int src_cnt, size_t len, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_xor_val)(
                struct dma_chan *chan, dma_addr_t *src, unsigned int src_cnt,
                size_t len, enum sum_check_flags *result, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_pq)(
                struct dma_chan *chan, dma_addr_t *dst, dma_addr_t *src,
                unsigned int src_cnt, const unsigned char *scf,
                size_t len, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_pq_val)(
                struct dma_chan *chan, dma_addr_t *pq, dma_addr_t *src,
                unsigned int src_cnt, const unsigned char *scf, size_t len,
                enum sum_check_flags *pqres, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_interrupt)(
                struct dma_chan *chan, unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_dma_sg)(
                struct dma_chan *chan,
                struct scatterlist *dst_sg, unsigned int dst_nents,
                struct scatterlist *src_sg, unsigned int src_nents,
                unsigned long flags);

        struct dma_async_tx_descriptor *(*device_prep_slave_sg)(
                struct dma_chan *chan, struct scatterlist *sgl,
                unsigned int sg_len, enum dma_transfer_direction direction,
                unsigned long flags, void *context);
        struct dma_async_tx_descriptor *(*device_prep_dma_cyclic)(
                struct dma_chan *chan, dma_addr_t buf_addr, size_t buf_len,
                size_t period_len, enum dma_transfer_direction direction,
                unsigned long flags);
        struct dma_async_tx_descriptor *(*device_prep_interleaved_dma)(
                struct dma_chan *chan, struct dma_interleaved_template *xt,
                unsigned long flags);
        int (*device_control)(struct dma_chan *chan, enum dma_ctrl_cmd cmd,
                unsigned long arg);

        enum dma_status (*device_tx_status)(struct dma_chan *chan,
                                            dma_cookie_t cookie,
                                            struct dma_tx_state *txstate);
        void (*device_issue_pending)(struct dma_chan *chan);
        int (*device_slave_caps)(struct dma_chan *chan, struct dma_slave_caps *caps);
```

这些callback function是怎样被调用的?

---

struct dma_chan * dma_request_channel(const dma_cap_mask_t *mask,

dma_filter_fn fn, void *fn_param);

struct dma_chan *dma_get_slave_channel(struct dma_chan *chan)

struct dma_chan *dma_get_any_slave_channel(struct dma_device *device);

调用

```
        int (*device_alloc_chan_resources)(struct dma_chan *chan)
```

---

void dma_release_channel(struct dma_chan *chan);

调用

```
void (*device_free_chan_resources)(struct dma_chan *chan);
```

---

inline void dma_async_issue_pending(struct dma_chan *chan)

```
static inline void dma_async_issue_pending(struct dma_chan *chan)
{
        chan->device->device_issue_pending(chan);
}
```

调用

```
void (*device_issue_pending)(struct dma_chan *chan);
```

```
static inline enum dma_status dmaengine_tx_status(struct dma_chan *chan,
        dma_cookie_t cookie, struct dma_tx_state *state)
{
        return chan->device->device_tx_status(chan, cookie, state);
}
```

```
static inline enum dma_status dma_async_is_tx_complete(struct dma_chan *chan,
        dma_cookie_t cookie, dma_cookie_t *last, dma_cookie_t *used)
{
        struct dma_tx_state state;
        enum dma_status status;

        status = chan->device->device_tx_status(chan, cookie, &state);
        if (last)
                *last = state.last;
        if (used)
*used = state.used;
        return status;
}
```

调用

```
        enum dma_status (*device_tx_status)(struct dma_chan *chan,
                                            dma_cookie_t cookie,
                                            struct dma_tx_state *txstate);
```

---

int dma_get_slave_caps(struct dma_chan *chan, struct dma_slave_caps *caps)

调用

```
int (*device_slave_caps)(struct dma_chan *chan, struct dma_slave_caps *caps);
```

---

struct dma_async_tx_descriptor *dmaengine_prep_slave_single(

    struct dma_chan *chan, dma_addr_t buf, size_t len,

    enum dma_transfer_direction dir, unsigned long flags)

struct dma_async_tx_descriptor *dmaengine_prep_slave_sg(

    struct dma_chan *chan, struct scatterlist *sgl,   unsigned int sg_len,

    enum dma_transfer_direction dir, unsigned long flags)

调用

```
        struct dma_async_tx_descriptor *(*device_prep_slave_sg)(
                struct dma_chan *chan, struct scatterlist *sgl,
                unsigned int sg_len, enum dma_transfer_direction direction,
                unsigned long flags, void *context);
```

Note:

```
static inline int dmaengine_slave_config(struct dma_chan *chan,
                                          struct dma_slave_config *config)
{
        returndmaengine_device_control(chan, DMA_SLAVE_CONFIG,
                        (unsigned long)config);
}
```

调用

```
        int (*device_control)(struct dma_chan *chan, enum dma_ctrl_cmd cmd,
                unsigned long arg);
```