dma相关api分为两部分

1. 分配dma用的memory (dma buffer)
2. dma driver

对于1，就是类似如下kernel apis

```
void *
dma_alloc_coherent(struct device *dev, size_t size,
                   dma_addr_t *dma_handle, gfp_t flag)

void
dma_free_coherent(struct device *dev, size_t size, void *cpu_addr,
                  dma_addr_t dma_handle)

struct dma_pool *
    dma_pool_create(const char *name, struct device *dev,
                    size_t size, size_t align, size_t alloc);

void *dma_pool_alloc(struct dma_pool *pool, gfp_t gfp_flags,
                     dma_addr_t *dma_handle);
```

etc.

Documentation/DMA-API.txt对此有描述。

对于2，则是dma hardware driver的framework，dmaengine.

dmaengine就象一个具体dma hardware driver的pure virtual base class，定义了child class需要实现的interfaces。各个具体dma driver的任务就是实现这些interfaces.而dma user一般只会与dmaengine的exported apis打交道。

Documentation/dmaengine.txt对此有描述。

要使用dma functionality的kernel code 通过1的api分配memory,而通过2来操作dma hardware。