- driver_usb
  - usb-hid-app
    - USBSystem.init().init_probe().drive_all();
  - 抽象
    - mod
      - OSAbstractions
        - VirAddr：虚拟地址类型
        - DMA：内存分配器
        - PAGE_SIZE：页面大小
        - 提供dma_alloc接口
        - 提供send_event接口，暂时没有实现
      - HALAbstractions
        - 提供force_sync_cache接口
    - dma
      - 定义dma相关的结构体和方法
    - event
  - 胶水
    - DriverIndependentDeviceInstance结构体
      - slotid：设备插槽号
      - configuration_val：当前配置值，默认是1
      - interface_val：当前接口值，默认是0
      - current_alternative_interface_value：备用接口值，默认也是0，感觉没什么用
      - descriptors: 使用原子引用计数和MightBeInited结构体保护的TopologicalUSBDescriptorRoot
      - controller：ControllerArc类型，是一个对于控制器的引用，便于通过DriverIndependentDeviceInstance来访问控制器的功能
    - URB
      - CompleteCode：完成代码，代表USB请求的完成状态
      - _phantom_data：标记与泛型O相关的类型信息。
  - 主机（xhci）层驱动
    - mod
      - USBHostSystem结构体，代表主机层
        - config：原子引用计数和无中断自旋锁保护的USBSystemConfig
        - controller：ControllerArc类型，对于控制器的引用，便于通过USBHostSystem类型来访问主机控制器
        - new()：调用XHCI的new()创建一个新的控制器，加上config，创建USBHostSystem实例
        - init()：调用controller的init()方法进行初始化
        - probe()：调用controller的probe()探测主机连接的所有设备，然后创建一个DriverIndependentDeviceInstance驱动无关的设备实例，最后传给一个consumer函数，这个函数目前有一个情况是把这个实例push给USBSystem。
        - control_transfer：调用controller的control_transfer()，传入插槽号和urb_req，这个urb_req定义在usb驱动层中，ControlTransfer类型。
        - configure_device：调用controller的configure_device()，传入插槽号和urb_req，这个urb_req定义在usb驱动层中，Configuration类型。
        - urb_request：传入一个URB类型参数，根据参数对应的操作类型来调用控制器的不同功能。
          - Control
            - 调用控制器的controll_transfer()
          - Bulk
            - 批量传输，暂未实现
          - Interrupt
            - 调用控制器的interrupt_transfer
          - ConfigureDevice
            - 调用控制器configure_device
          - ExtraStep
            - 调用控制器的extrastep
        - tock：传入一个todolist的list，调用urb_request来完成所有的URB请求
    - data_structures
      - mod定义了一个MightBeInited类型，只有两类Inited(T)和Uninit,
      - host_controller
        - mod
          - 定义Controller trait
            - new()创建一个控制器实例
            - init()初始化一个控制器实例
            - probe()探测控制器中的设备
            - control_transfer()控制传输
            - interrupt_transfer()中断传输
            - configure_device()配置设备
            - extra_step()额外步骤
            - device_slot_assignment()分配设备插槽
            - address_device()给设备分配地址
            - control_fetch_control_point_packet_size()传入插槽号，获取控制端点数据包大小
            - set_ep0_packet_size()设置端点0的数据包大小
          - 定义ControllerArc为原子引用计数和无中断自旋锁保护的实现了控制器trait的类型。
        - xhci是对控制器功能的具体实现
          - mod
          - context
          - envent_ring
          - ring
  - usb层驱动
    - descriptor
    - drivers
      - driver_api
        - USBSystemDriverModule trait
        - USBSystemDriverModuleInstance trait
      - DriverContainers结构体
        - drivers：封装USBSystemDriverModule的Box引用的数组
        - new()
        - load_driver()：push一个驱动模块到数组中
        - create_for_device()：遍历drivers中加载的所有驱动模块，调用模块的should_active方法，判断是否应该active。对于应该激活的模块，返回一个驱动设备实例。
          - 对于每一个驱动设备实例，调用prepare_for_drive()来获取驱动需要传输的URB的list。修改传入的preparing_list参数用于保存。
          - 返回驱动设备实例数组
    - operation
      - Configuration枚举
        - SetupDevice(&'a TopologicalUSBDescriptorConfiguration)设置设备操作
        - SwitchInterface(InterfaceNumber, AltnativeNumber),切换接口操作
      - ExtraStep枚举
    - transfer
    - universal_driver
    - mod
      - USBDriverSystem结构体
        - config：原子引用计数和无中断自旋锁保护的USBSystemConfig结构体实例
        - managed_modules：DriverContainers类型，本质上是USBSystemDriverModule的Box引用的一个数组
        - driver_device_instances：使用原子引用计数和无中断自旋锁保护的DriverIndependentDeviceInstance数组
        - new()：根据USBSystemConfig来创建一个USBDriverSystem子系统实例
        - init()：调用managed_modules的load_driver方法，加载一个具体驱动模块的Box引用到managed_modules中。
        - init_probe()
          - 传入一个驱动无关的设备列表，这个设备列表是主机层探测出的一个列表。
          - 传入一个preparing_list来保存生成的URB请求。
          - 遍历所有设备，调用managed_modules的create_for_device，来为设备产生URB。
          - 收集所有设备到USBDriverSystem的driver_device_instances中。
        - tick()
          - 遍历driver_device_instances列表，调用每个设备实例的gather_urb
          - 对于每个URB请求，调用其set_sender请求，设置为对应的驱动设备实例。
          - 最后将所有驱动设备的URB请求都收集到一个向量中。
    - urb
      - 定义URB结构体
        - device_slot_id
        - operation枚举
          - ExtraStep
          - Control
          - Bulk
          - Interrupt
          - Isoch
        - sender：Option和原子引用计数以及无中断自旋锁封装的USBSystemDriverModuleInstance实例
  - 库（向外暴露）
    - 结构体USBSystemConfig
      - base_addr：操作系统定义的USB系统基地址（虚拟地址）
      - irq_num：USB系统使用的中断号
      - irq_priority：USB系统使用的中断优先级
      - os：提供平台抽象的实例
    - 结构体USBSystem
      - platform_abstractions：平台抽象实例
      - config：使用原子引用计数和无中断自旋锁保护的USBSystemConfig
      - host_driver_layer：主机驱动层
      - usb_driver_layer：usb驱动层
      - driver_independent_devices：胶水层定义的驱动无关设备组成的一个数组
      - new()：根据USBSystemConfig的实例新建USBSystem
      - init()：调用主机驱动层的init()和USB驱动层的init()
      - init_probe()：调用主机驱动层的probe()，获取主机连接的USB设备。调用new_device()把每个设备添加到驱动无关设备数组中。调用USB驱动层的init_probe()生成一个preparing_list。最后调用主机驱动层的tock()。来传输这个preparing定义的所有请求。
      - driver_active()：无操作，供扩展
      - drop_device()：无操作，供扩展
      - new_device：检查DriverIndependentDeviceInstance的描述符字段，判断是否初始化。
        - 已初始化
          - 调用主机驱动层的urb_request()，发送配置设备的URB
        - 未初始化
          - 分配DMA缓冲区，发送控制请求以获取设备描述符，解析设备描述符，获取配置描述符数目，解析配置描述符，保存在设备实例的descriptor字段中
      - drive_all：轮询USB驱动层产生的所有tick。并且调用主机驱动层的tock()，进行响应。

Presented with xmind