前言

本文参考了 XHCI on OS Dev 以及《USB: The Universal Series Bus》。代码存储于 `github.com/juruohejiarui/VCPP-2.git` 中的 `VOS/kernel/hardware/USB`.
需要先简单了解 PCIe 的数据结构和枚举、检测 UEFI 提供的 PCIe 信息的方法。
可以遵循下列步骤设置你的 XHCI 控制器：

1. 找到所有 XHCI 的数据结构
2. 获取所需要的信息，建立所需要的额外的数据结构
3. 重置控制器，并设置控制器
4. 启动控制器
5. 开启监测进程

数据结构和注释

XHCI 控制器有大量的寄存器组和描述，需要一定的耐心。但是我们不需要全部了解，只需要知道一些基本的，可以用于设备枚举，检测和通信的寄存器就可以了。但是为了查阅方便，这里我还是几乎完全摘抄了《USB: The Universal Series Bus》对于每一个描述符的解释。当我们需要某些功能的时候再回来查阅表格即可。如果学习过程中发现一些不懂得名词，可以先把它记下来，之后遇到了定义或解释的时候再返回来看，如果之后都没有出现，那么就可以认为它并不重要。

PCIe

从 UEFI 提供的 `MCFG` 表中，可以读出一个 PCIe 数据表头的结构，我们称之为 `PCIeConfig`，对于 XHCI 设备，其结构一般如下：

Offset	Size(Byte)	Name	Description
0x0	2	Vendor ID	制造商 ID
0x2	2	Device ID	制造商申请的设备特定的 ID
0x4	2	Command	用于操作该 PCIe 设备的寄存器
0x6	2	Status	该设备的状态
0x8	1	Revision ID	我也不懂，应该不太重要，可以忽略
0x9	1	Prog IF	Program Interface Byte, 只读，用于寄存器级别下指定该设备的类别
0xa	1	Subclass	子类 ID
0xb	1	Class Id	类 ID
0xc	1	Cache Line Size	我也不懂，应该不太重要，可以忽略
0xd	1	Latency Timer	用于 PCI，PCIe 不支持，固定为 0

Offset	Size(Byte)	Name	Description
0xe	1	Header Type	指示该表头剩余内容的结构
0xf	1	BIST	Built-in self test, 设备的内建自检状态和控制寄存器
0x10	4	Bar0	Base Address 0
0x14	4	Bar1	Base Address 1
0x18	4	Bar2	Base Address 2
0x1c	4	Bar3	Base Address 3
0x20	4	Bar4	Base Address 4
0x24	4	Bar5	Base Address 5
0x28	4	Cardbus CIS Pointer	指向 Cardbus 结构的指针，该值被 PCI 和 Cardbus 共享（机翻）
0x2c	2	Subsystem Vendor ID	子系统的 Vendor ID, 不重要
0x2e	2	subsystem ID	子系统 ID，不重要
0x30	4	Expansion ROM base address	不懂，应该不太重要
0x34	1	Capabilities Pointer	使能寄存器的偏移量
0x35	7	Reserved	保留位置，不能读写
0x3c	1	Interrupt Line	用于指定该设备连接到 PIC (不是 I/O APIC) 的 (IRQ0,IRQ1...IRQ15) 哪个（些）引脚, `0xff` 则表示无连接
0x3d	1	Interrupt PIN	用于指定该设备使用哪个中断引脚，`0x0` → does not use, `0x1` → `INTA`, `0x2` → `INTB`, `0x3` → `INTC`, `0x4` → `INTD`
0x3e	1	Min Grant	用于 PCI ，PCIe 下不可用，固定为 0
0x3f	1	Max Latency	用于 PCI ，PCIe 下不可用，固定为 0

Class Id, subClass, Prog IF 三者共同确认一个设备的类别，网上有对照表，这里就不粘出来了。对于 XHCI 设备，Class Id = `0xC`, Subclass = `0x3`, Prog IF = `0x30`

Header Type 结构

Bit 7	Bit 0 : 6
该设备是 (1)/否 (0) 为多功能 (multi-function) 设备	表头剩余结构的类型

对于 Bit 0 : 6 ，只有 `0x0, 0x1, 0x2` 三种取值，分别表示该 PCIe 设备是一个 General Device, PCI-to-PCI bridge, PCI-to-Cardbus bridge ，对于 XHCI 设备，其 Header Type 值一定为 `0x0` 或 `0x80` 。而上述展示的 PCIe 表结构就是 Bit 0 : 6 = `0x0` 时的结构，对于其他情况，可以参考 PCI on OS Dev 。

Capability Registers

可以翻译为使能寄存器组，这些寄存器均为只读 (RO) 寄存器。存储了大量的控制器支持信息。其基地址可以使用 `PCIeConfig` 的 bar0, bar1 组合获取，计算方式如下：

```
// 将两个地址的值拼接起来，然后对 4K 下取整对齐
addr=(bar0 | (bar1 << 32)) & (~((1 << 12) - 1))
```

这个 `addr` 可以记录为 baseAddr ，稍后依然会用到。接下来直接列出这些寄存器的信息：

Offset	Size(Byte)	Name	Description
0x0	1	CapLen	使能寄存器组的大小
0x1	1	RSVD	保留
0x2	2	hciVersion	接口的版本数字，不需要关注
0x4	4	hcsParams1	结构化的参数组 #1
0x8	4	hcsParams2	结构化的参数组 #2
0xc	4	hcsParams3	结构化的参数组 #3
0x10	4	hccParams1	使能参数组 #1
0x14	4	dbOff	Doorbell 寄存器组的偏移，对 4 对齐（相对于 baseAddr）
0x18	4	rtsOff	运行时寄存器空间的偏移，对 64 对齐（相对于 baseAddr）
0x1c	4	hccParams2	使能参数组 #2

上面的表中提到了五个参数组，实际上就是将一些值比较小的信息拼接起来，节省空间，接下来给出每个参数组的具体内容。

hcsParams 1, 结构化的参数组 #1

Bit	Name	Description
0:7	MaxSlots	最大插槽的编号
8:18	MaxIntrs	最大的中断编号
19:23	Reserved	保留
24:31	MaxPorts	最大端口的编号

这三个最大值可以用于枚举端口和中断，但注意，可用的端口和选择使用的中断可能并不连续。

hcsParams 2, 结构化的参数组 #2

Bit	Name	Description
0:3	Isochronous Scheduling Threshold (IST)	告诉软件它可以在多远的帧中修改 TRB（传输请求块）并且仍能正确执行
4:7	Event Ring Segment Table Max (ERST Max)	Event Ring Segment Table Base Size registers (稍后会有解释) 可以支持的最大值
8:20	Reserved	保留
21:25	Max Scractchpad Buffer (High 5 bits)	
26	Scratchpad Restore (SPR)	表示是否支持暂存器保存/恢复缓冲区
27:31	Max Scractchpad Buffer (Low 5 bits)	

将 Max Scractchpad Buffer 拼接起来可以得到软件需要申请的用于暂存器缓冲区的内容大小，如果该值为 0，那么就不需要申请内容，更多要求则和接下来提到的 pageSize 的值有关，但是在本人的机子上，这个值都是 0。

hcsParams 3, 结构化的参数组 #3

Bit	Name	Description
0:7	U1 Device Exit Latency	电源状态从 U1 到 U0 所需要的微秒数 ($10^{-6}s = 1\mu s$)
8:15	Reserved	保留
16:31	U2 Device Exit Latency	电源状态从 U1 到 U0 所需要的微秒数

hccParams 1, 使能参数组 #1

Bit	Name	Description
0	64-bit Addressing Capability (AC64)	该 XHCI 控制器是 (1) 否 (0) 可以使用 64-bit 地址，如果不能，那么接下来使用的所有地址，都会忽略高 32 位。
1	BW Negotiation Capability (BNC)	用于指示该控制器是 (1) 否 (0) 实现了 Bandwidth Negotiation
2	Context Size (CSZ)	指示上下文的大小 1: 64-byte; 0: 32-byte 。该位对 Stream Contexts 不生效。
3	Port Power Control (PPC)	1: 所有的端口永远处于通电状态; 0: 每个端口可以自行控制供电状态

Bit	Name	Description
4	Port Indicators (PIND)	用于指示端口指示灯的控制。0: 控制不可用; 1: 参照端口寄存器的 Bit 14:15 来获取更多信息
5	Light HC Reset Capability (LHRC)	指示操作寄存器组中的 usbCmd 寄存器的 Light Host Controller 功能是 (1) 否 (0) 可用
6	Latency Tolerance Messaging Capability (LTC)	表示该控制器是 (1) 否 (0) 支持传递延迟容忍消息 (Latency Tolerance Messageing, LTM)
7	No Secondary SID Support (NSS)	**是 (0) 否 (1)** 支持 Secondary Stream IDs.
8	Parse All Event Data (PAE)	1: 当转移到下一个传输描述符的时候，所有的 TRB 都会被解析；0: 不会。检测到一个 Short Packet 的时候会使用这个位
9	Stopped - Short Packet Capability (SPC)	该控制器是 (1) 否 (0) 可以发出短数据包停止代码 (Stopped-Short Packet Completion code)
10	Stopped EDTLA Capability (SEC)	该控制器是 (1) 否 (0) 在 Stream Context 中实现了 Stopped EDTLA 区域，对于较新的硬件，这一位都是 1。
11	Continuous Frame ID Capability (CFC)	该控制器是 (1) 否 (0) 能够匹配连续 ISO 传输描述符的 ID。
12:15	Maximum Primary Stream Array Size (MaxPSASize)	允许的最大 Stream Array 大小
16:31	XHCI Extended Capabilities Pointer (xECP)	0: 该控制器不存在拓展使能描述; >0: 拓展使能描述的偏移（相对于 baseAddr ）

实际上的 Stream Array 最大大小需要使用如下方式计算:

$$\text{maxSize} = 2^{\text{MaxPSASize}+1}$$

对于拓展时能描述的偏移实际上是:

$$\text{offset of extended Capabilites} = \text{baseAddr} + 4 \times \text{xECP}$$

hccParams 2, 使能参数组 #2

Bit	Name	Description
0	U3 Entry Capability	根 (root hub) 是否支持暂停完成通知 (Suspend Completion notification)
1	ConfigEP Command Max Exit Latency too Large (CMC)	是 (1) 否 (0) 可以生成最大退出潜伏期过长的兼容错误 (Exit Latency too Large compatibility error)，该位被操作寄存器的 usbCmd 的 CME 位使用。
2	Force Save Context Capability (FSC)	Save State 命令是 (1) 否 (0) 应当将所有的 cached Slot, End Point, Stream 和其他上下文存储到内存中
3	Compliance Transition Capability (CTC)	该控制器是 (1) 否 (0) 支持 Compliance Transition Enabled 位
4	Large ESIT Payload Capability (LEC)	是 (1) 否 (0) 支持大于 48Kb 的 ESIT Payloads
5	Configuration Information Capability	是 (1) 否 (0) 支持输入控制上下文块 (Input Control Context block) 中的 Configuration Value, Interface Number 和 Alternate settings 域的拓展配置信息
6:31	Reserved	保留

Extended Capabilites List

除了上面的 Capability Registers，XHCI 还提供了链表结构的拓展使能列表，和 PCIe 非常类似。每个拓展描述符都长下面这样：

Bit	Name	Description
0:7	ID	RO, 该拓展描述符的 ID
8:15	Next	RO, 下一个拓展描述符相对于当前描述符的偏移，计算方式如下： next offset = current offset + 4 × Next
n	Varies	每个 ID 特定的内容

下面给出一些常用的 ID 对照表：

ID	Name	Description	Size(Byte)
0	Reserved		

ID	Name	Description	Size(Byte)
1	USB Legacy Support	提供操作系统启动前，既 UEFI 或 BIOS 的支持。如果这个拓展描述符是 USB Legacy Support，那么它一定是第一个拓展描述符。(书上是这么说的，但是貌似并不是这样的)	8
2	Supported Protocol	该控制器支持的其中一种协议，每个控制器至少有一个这种拓展描述符，并且可能有多个。	16
3	Extened Power Management	拓展的电源管理，至少有一个。	n
4	I/O Virtualization	硬件虚拟化支持	最高 1280
5	Message Interrupt	信息中断的支持	n
6	Local Memory	本地内存支持	最高 4TB
7-9	Reserved		
10	USB Debug Capability	调试功能支持	56
11-16	Reserved		
17	Extended Message Interrupt	拓展的信息中断的支持	
18-191	Reserved		
192-255	Vendor Specific	特定的 Vendor 所使用的拓展使能信息	

VMware 虚拟机的 XHCI 控制器似乎并不存在 USB Legacy Support 描述符。
每个 XHCI 控制器，至少拥有一个 Message Interrupt 或者 Extended Message Interrupt 描述符。一般而言，有 Extended Message Interrupt 则使用 Extend Message Interrupt，否则使用 Message Interrupt 。

XHCI USB Legacy Support

整一个描述符可以使用一个 64 位整数存储，但是为了便于展示，将其分成两个 32 位整数。
对于第一个整数:

Bit	Name	Description
0:7	ID	RO, = 0x1
8:15	Next	RO
16	HC BIOS Owned Semaphore	R/W, Default = 0. BIOS 将此位设置为 1 来指示 BIOS 占用了该控制器。
17:23	Reserved and preserved	R/W

Bit	Name	Description
24	System Software Owned Semaphore	R/W, Default = 0，操作系统将此位设置为 1 来获取这个控制器的所有权，设置之后，待到 Bit 16 变为 0 即表示所有权获取成功
25:31	Reserved and preserved	R/W

对于要求 Reserved and preserved 的位置，需要每次写入的之前先把这些 Bit 的值读出来，或上要写入的其他位置的值，然后再将整个整数写入。
第二个整数的值被 BIOS 使用，操作系统软件无需关注。对于这个拓展描述符，唯一要做的就是将第一个整数的 Bit 24 设置为 1 ，然后等待 Bit 16 变为零即可。

XHCI USB Supported Protocol Capability

本拓展描述符用于描述控制器支持的其中一种协议，以及对应的端口序列。类似的，我们用 4 个 32 位整数描述这个描述符：
对于第一个整数：

Bit	Access	Description
0:15	RO	ID 和 Next
16:23	RO	Minor Revision Number, “USB 3.0” 中的 0
24:27	RO	Reserved
28:31	RO	Major Revision Number, “USB 3.0” 中的 3

注意，Minor Revision Number 和 Major Revision Number 均为整数，而不是字符。在 VMware 的 USB 3.1 控制器中，其中一个拓展描述符有 Major Revision Number = 3, Minor Revision Number = 1 ，而另一个拓展描述符有 Major Revision Number = 2, Minor Revision Number = 0 ，也就是说 VMware 的 XHCI 支持 USB 3.1 和 USB 2.0 两种协议。
第二个整数可以视为长度为 4 的字符串，其值一定为 “USB” 。
第三个整数存储了支持该协议的端口，

Bit	Access	Description
0:7	RO	Compatible Port Offset，兼容该协议的端口的偏移，也就是从第几个端口开始支持这种协议
8:15	RO	Compatible Port Count，兼容该协议的端口数量
16:27	RO	Protocol Defined，对于 USB 3.0 ，该段表示 Hub Depth；对于 USB 2.0，Bit 17 表示该控制器是 (1) 否 (0) 只支持高速设备，剩余位用于表示 Hub Depth。详细内容可以查询 XHCI 技术手册的 Section 7.2.2.1.3

Bit	Access	Description
28:31	RO	Protocol Speed ID Count，> 0：表示用于定义被该控制器支持的速度类型的 PSI 结构的 32 位整数的个数。

对于第四个整数，只有 XHCI 1.1 或更新版本才有定义，否则所有位均为 Reserved and preserved 。

Bit	Access	Description
0：4	RO	Slot Type ，插槽类型，目前只支持 0
5:31	RO	Reserved and preserved

Operational Registers，操作寄存器组

这一组寄存器用于整个控制器的操作，包括重置，上电/断电等等。可以使用 opRegAddr = baseAddr + capRegs.capLen 获取这组寄存器的地址。若寄存器为 32 位，那么只能用读写 32 位整数的方法对其操作；只可以使用 64 位整数的操作方法对 64 位寄存器进行操作。这里涉及到 C 语言编译出的汇编码的限制，使用优化的时候需要注意。

Offset	Size(Byte)	Name	Description
0x0	4	usbCmd	用于操作控制器的寄存器
0x4	4	usbState	用于指示控制器状态的寄存器
0x8	4	pageSize	用于指示可用的地址的页大小的寄存器
0xC	8	Reserved	保留
0x14	4	devNotifCtrl	Device Notification Control，用于控制控制器的报告内容的寄存器
0x18	8	cmdRingCtrl	Command Ring Control
0x20	8	Reserved	保留
0x30	8	devCtxBaseAddr	Device Context Base Address，用于保存上下文结构的基地址
0x38	4	config	Configuration，配置
0x3c	964	Reserved	保留
0x400	$16n$	PortRegs	端口的寄存器组集合

下一个小结会解释端口寄存器组的数据结构。

usbCmd，对控制器的命令

对这个寄存器写入之后不一定立刻有效果。因此写入之后需要保证该位被成功设置。

Bit	Access	Name	Description
0	R/W	Run/Stop (RS)	Run(1);Stop(0)，写入 0 之后，需要等到 usbState 中的 usbHalted 位被成功置位才算完成操作。
1	R/W	Host Controller Reset (HCRST)	重置。将操作寄存器组的所有寄存器设为默认值，将所有的 Transaction 设置为停止状态 (Halt)。对 USB2.0 下游设备无效；使 USB3.0 设备进行热重置 (Hot Reset) 或温重置 (Warm Reset) 。需要保证 usbHalted 位在重置之前已经被设置为 1 。重置完成之后这一位会被复位成 0 。
2	R/W	Interrupter Enable (INTE)	是 (1) 否 (0) 能让中断器移动到下一个中断临界点 (Interrupt threshold)，= 0：禁用中断
3	R/W	Host System Error Enable (HSEE)	是 (1) 否 (0) 允许当 usbState 的 HSE 位被设为 1 的时候触发中断
4:6	R/W	Reserved and Preserved	保留
7	R/W or RO	Light Host Controller Reset (LHCRST)	如果 hccParams1 的 LHRC 位为 1 ，那么为 R/W，可以用于重置灯组寄存器，重置完成之后被复位为 0 ；如果为 0 ，那么这一位为 RO 位
8	R/W	Controller Save State (CSS)	用于指示是否将状态存储到操作系统申请得到的 Scratch buffers 中。在设置为 1 之前，需要保证 usbHalted 位为 1 ，保存过程中，usbState 中的 Save State Status 位会被设置为 1 。

Bit	Access	Name	Description
9	R/W	Controller Restore State (CRS)	用于指示是否从 Scratch buffer 中恢复状态，在设置为 1 之前，需要保证 usbHalted 位为 1 。类似的，用 usbState 中的 Restore State Status 位表示是否恢复完成，恢复时被设置为 1。
10	R/W	Enable Wrap Event (EWE)	是 (1) 否 (0) 在 MFINDEX 寄存器从 `0x3fff` 变为 `0x0000` 时候触发中断
11	R/W	Enable U3 MFINDEX Stop (EU3S)	是 (1) 否 (0) 在通电状态处于 U3 的时候停止 `MFINDEX` 寄存器
12	R/W	Stopped Short Packet Enable (SPE)	是 (1) 否 (0) 产生发出短数据包停止代码 (Stopped-Short Packet Completion code)
13	R/W	CEM Enable (CME)	是 (1) 否 (0) 可以在 Configuration Endpoint Command 指令被接受并产生错误的时候，返回最大退出潜伏期过长的兼容错误 (Exit Latency too Large compatibility error)
14:31	R/W	Reserved and preserved	保留

usbState，控制器的状态

这里要解释一下 WC 即 Write Clear 的意思：当某一位被写入 0 的时候，不会改变该位的值，当写入 1 的时候，该位变为 0 。

Bit	Access	Name	Description
0	RO	HCHalted	控制器是否因为 RS 位被设置，或者控制器产生错误而停止。
1	R/W	Reserved and preserved	保留
2	R/WC	Host System Error (HSE)	当控制器产生严重错误的时候被设置为 1 ，否则为 0 。

Bit	Access	Name	Description
3	R/WC	Event Interrupt (EINT)	当任何一个中断器的 Interrupt Pending 位被设置的时候，这一位被设置为 1，否则为 0 。
4	R/WC	Port Change Detected (PCD)	当控制器的任何一个端口的连接状态位从 0 变为 1 的时候，这一位被设置为 1，否则为 0 。
5:7	R/W	Reserved and preserved	保留
8	RO	Save State Status (SSS)	当保存状态的操作处于进行时，这一位被设置为 1，保存完成之后会被设置为 0 。
9	RO	Restore State Status (RSS)	当恢复状态的操作处于进行时，这一位被设置为 1，恢复完成之后会被设置为 0 。
10	R/WC	Save/Restore Error (SRE)	表示保存状态/恢复状态的操作是 (1) 否 (0) 产生错误。
11	RO	Controller Not Ready (CNR)	如果被设置，则禁止对操作寄存器组（除了 $usbState）或 Doorbell 寄存器组进行任何写入操作。当处于“重置中”时，这一位会被设置。
12	RO	Host Controller Error (HCE)	表示是 (1) 否 (0) 发生外部错误，需要重置和重新初始化控制器。
13:31	R/W	Reserved and preserved	保留

pageSize，该控制器支持的页大小

该寄存器的 Bit 16:31 为保留位，对于 Bit 0:15 ，如果其中的 Bit x 被设置为 1 ，则表示该控制器支持大小为 2^{x+12}B 的物理页。在本人的机子上，均有 pageSize $\cap$ `0xffff` $= 0$ 。可以使用如下计算方法直接获得实际大小:

```
size = (pageSize & 0xfffful) << 12
```

devNotifCtrl，控制器接受通知包的控制寄存器

该寄存器的 Bit 16:31 为保留位，对于 Bit 0:15 ，如果其中的 Bit x 被设置为 1 ，则表示激活第 x 种通知类型。

cmdRingCtrl，指令环的控制寄存器

Bit	Access	Name	Description
0	R/W	Ring Cycle State (RCS)	告知控制器携带一个设置位 (set bit) 或者一个清理位 (clear bit) 来开始处理指令环
1	R/WC	Command Stop (CS)	告知控制器是否停止处理指令环。在处理完成当前指令之后会停止处理指令环，然后在事件环中放入一个任务环停止事件 (Command Ring Stopped event)
2	R/WC	Command Abort (CA)	直接停止处理指令环，然后放入一个任务环停止事件 (Command Ring Stopped event)
3	RO	Command Ring Running (CRR)	表示当前是否在处理指令环
4:5	R/W	Reserved and preserved	保留
6:63	R/W	Command Ring Pointer	指令环的地址的高 58 位，也就是将这个寄存器 `& (~0x3ful)` 即为指令环的地址

devCtxBaseAddr，设备上下文的基地址

Bit	Access	Name	Description
0:5	R/W	Reserved	必须为 0
6:63	R/W	Device Context Base Address Array Pointer	地址的高 58 位

实际上就是存储一个对 64 对齐的地址。

config，配置寄存器

Bit	Access	Name	Description
0:7	R/W	Max Device Slots Enabled (MaxSlotsEn)	设置激活的最大插槽 ID，只能在 RS = 0 的时候设置这个段。

Bit	Access	Name	Description
8	R/W	U3 Entry Enable (U3E)	表示控制器转移到 U3 状态的时候是否设置 PLC 标志位。hccParams0 的 Bit 0 指示是否支持这一位。
9	R/W	Configuration Information Enable (CIE)	是否初始化输入控制上下文块的 Configuration Value, Interface Number 和 Alternate Settings 域，hccParams0 的 Bit 5 指示是否支持这一位。
10:31	R/W	Reserved and preserved	保留

Port Registers, 端口状态和控制寄存器组

每一个端口寄存器组由 4 个 32 位整数描述。某些寄存器在 USB 2.0 和 3.1 下会有不同的含义。

Offset	Name	Description
0x0	stsCtrl	Port Status and Control, 用于端口的状态查询和控制，几乎每一个位都有作用
0x4	pwStsCtrl	Port Power Management Status and Control，用于端口电源的状态查询和控制
0x8	lkInfo	Port Link Info, 保存端口的链接信息
0xc	hwLPMCtrl	Port Hardware LPM Control

stsCtrl, 端口的状态和控制

下表展示了这个寄存器每一位的作用，对于某些位置，会有一些必要的补充说明。在 RS 被设置为 1 并且 usbHalted = 0 之后，这个寄存器才能正常使用。
PS: 标有下划线的位是仅适用于 USB 3.0 的位，对于 USB 2.0 协议的端口，这些位处于 Reserved and Zero'd 状态。

Bit	Access	Description
0	RO	Current Connect Status，当前连接状态，1: 有设备连接；0: 无设备连接。用于设备连接检测的关键位。
1	R/WC	Port Enabled(1)/Disabled(0)，端口激活状态。控制器在重置后，或者端口重置后会激活端口并设置该位，由于为 WC 位，因此操作系统不能自行激活端口，只能通过重置端口或者控制器重新激活。
2	R/W	Reserved and Zero'd

Bit	Access	Description
3	RO	Over-current Active，表示该端口是否处于过流激活状态 (Over-current Active condition)
4	R/W	Port Reset，和 usbCmd 的 HCRST 位功能类似。在不同协议下，重置操作略有不同。
5:8	R/W	Port Link State，链接状态，从 `0000b` 开始分别表示 U0, U1, U2, U3 。需要设置该寄存器的 Bit 16 来启用该位的写入。一般而言，需要让链接状态处于 U0 以正常使用。
9	R/W	Port Power，用于开启或关闭该端口的电源。hccParams1 的 Port Power 位指示该位是 (0) 否 (1) 可以正常使用。
10:13	RO	Port Speed，用于指示该端口连接设备的速度和对应的协议。接下来会提供一个参照表。
14:15	R/W	Port Indicator Control，指标控制，表示是否可以正常操作，接下来会提供一个参照表。hccParams1 的 PIND 指示该位是 (1) 否 (0) 可以正常使用。
16	R/W	Port Link State Write Strobe，用于指示对 Port Link State 的写入将会生效 (1)/被忽略 (0)。
17	R/WC	Connect Status Change，连接状态是否改变，用于设备连接检测的关键位。
18	R/WC	Port Enable/Disable Change，端口的激活状态是否被改变。
19	R/WC	Warm Port Reset Change，当端口为 USB 2.0 协议时，这一位被保留；当端口为 USB 3.0 时，若 Warm Port Reset 完成，这一位会被设置为 1
20	R/WC	Over-current Change，过流状态是否发生改变。
21	R/WC	Port Reset Change，当 Port Reset 位从 1 变为 0 的时候，这一位被设置为 1 。

Bit	Access	Description
22	R/WC	Port Link State Change，当 Link State 发生改变的时候，这一位被设置为 1 。
23	R/WC	Port Config Error Change，指示是 (1) 否 (0) 在配置链接伙伴 (link partner) 的时候发生错误。
24	RO	Cold Attach Status，表示连接的设备是否因为电量不足、错误的电源状态或者其他问题导致不能正常运作。
25	R/W	Wake on Connect Enable，是否在连接的时候产生对应的事件。
26	R/W	Wake on Disconnect Enable，是否在断开连接的时候产生对应的事件。
27	R/W	Wake on Over-current Enable，是否在进入过流状态的时候产生对应的事件。
28:29	R/W	Reserved and Zero'd
30	RO	Device Removable，当前设备是否是可移除的设备。
31	R/WC	Warm Port Reset，操作方式和 Port Reset 类似，用 Port Reset、Port Reset Change 和 Warm Reset Change 来指示 Warm Reset 是否完成。

接下来提供一个 Port Speed 的描述表：

Value	Speed	Bit Rate	Protocol
0	Undefined		
1	full-speed	12MB/s	USB 2.0
2	low-speed	1.5MB/s	USB 2.0
3	high-speed	480MB/s	USB 2.0
4	super-spped	5GB/s	USB 3.0

还有一个 Port Indicator Control 的参考表，由于不太懂英文的含义（不能理解为啥要用颜色来定义），因此直接直译书中原文：

Value	含义
00b	指标关闭
01b	琥珀色 (Amber)，通常表示存在错误，需要因此硬件注意。

Value	含义
10b	绿色 (Green)，一切正常。
11b	未定义

这里的指标指的是端口的指标，而不是端口上设备的指标。

pwStsCtrl，端口电源的状态和控制

对于使用 USB 2.0 的端口，使用下面的描述表：

Bit	Access	Description
0:2	RO	L1 Status，表示 L1 下的暂停 (suspend) 请求的状态。接下来会提供每个值对应的含义。
3	R/W	Remote Wake Enable，激活从 L1 状态的远程唤醒。
4:7	R/W	Host Initiated Resume Duration，若该段的值为 x ，则表示控制器初始化并退出 L1 状态后恢复所需要的时间为 $50 + 75x$ 。
8:15	R/W	L1 Device Slot，用于指示该端口的哪个设备槽用于产生 LMP Token packet，= 0 表示没有设备。
16	R/W	Hardware LPM Enable，= 1: 激活硬件控制的 LPM。当硬件 LMP 使能并不支持的时候，这个位被保留。
17:27	R/W	Reserved and preserved，保留
28:31	R/W	软件向这个段写入某些数字来进行特定的测试。查看技术手册 (xHCI ver 1.10 p.320) 来获取更多相关信息。

接下来是 L1 Status 的描述表：

Value	含义
0	Invalid，不合法，如果为 0，那么 L1 Status 应当被忽略。
1	Success，成功。端口成功转换到 L1 状态。
2	Not Yet，仍未转移到 L1 状态。
3	Not Supported，该端口不支持转移到 L1 状态。
4	Timeout or Error，超时或者发生错误。
5~7	Reserved

对于使用 USB 3.0 的端口，使用下面的描述符表：

Bit	Access	Description
0:7	R/W	U1 Timeout，表示在确定 U1 不活动并启动 U1 进入之前的延迟（以微秒为单位）。合法值为 `0x00...0x7f` 以及 `0xff`，若为 `0x00` 则表示不能进入 U1，否则表示微秒数。
8:15	R/W	U2 Timeour，表示在确定 U2 不活动并启动 U2 进入之前的延迟（以微秒为单位）。合法值范围和 U1 相同。若为 `0x00` 表示不能进入 U2，否则延迟的微秒数等于该段的值乘上 256
16	R/W	Force Link PM Accept，当值为 1，让端口去生成一个 Set Link Function LMP 。（不知道啥意思，这里粘出原文：tells the port to generate a Set Link Function LMP with the Force Link PM Accept Bit set.）
17:31	R/W	Reserved and preserved，保留

lkInfo，链接信息

若端口为 USB 2.0 那么这个寄存器为 Reserved and preserved 。
若端口为 USB 3.0 那么使用下面的描述表：

Bit	Access	Description
0:15	RO	Link Error Count，检测到的链接错误的的数量。
16:19	RO	Rx Lane Count (RLC)，表示端口协商的接收 (Receive) 通道数减去 1
20:23	RO	Tx Lane Count (RLC)，表示端口协商的传输 (Transmit) 通道数减去 1
24:31	R/W	Reseerved and preserved，保留

hwLPMCtrl，硬件的 LPM 控制寄存器

若端口为 USB 2.0 那么使用下面的描述表：

Bit	Access	Description
0:1	RWS	Host Initiated Resume Duration Mode (HIRDM)，必须为 0 或 1 。
2:9	RWS	L1 Timeout，表示超时时间除以 128
10:13	RWS	Best Effort Service Latency Deep (BESLD)，表示接收设备在 U2 退出后要等待多长时间。
14:31	RWS	Reserved and preserved，保留

若端口为 USB 3.0 那么这个寄存器为 Reserved and preserved 。

Runtime Registers，运行时寄存器组

可以使用 rtRegAddr = baseAddr + capRegs.rtsOff 获取这组寄存器的位置。本组寄存器包含了当前所在的微帧 (microframe) 索引，以及至多 1024 个中断寄存器组，实际中断个数可以通过 hcsParams 的 maxIntr 段减去 1 得知。至于微帧的定义，可以参考稍后的内容。

Offset	Size(Byte)	Name	Description
0x0000	4	Microframe Index Register	当前所在的微帧的编号。如果 RS = 1，控制器每隔 125ms 就会让该数值递增一次，也就是在每个开始处理每个微帧之前就会递增一次。
0x0004	28	Reserved and Zero'd	
0x0020	32	Interrupter Register Set 0 (IR0)	第 0 个中断寄存器组
0x0040	32	Interrupter Register Set 1 (IR1)	第 1 个中断寄存器组
...	...	...	
0x8000	32	Interrupter Register Set 1023 (IR1023)	第 1023 个中断寄存器组

Interrupter Registers，中断寄存器组

第 0 个中断被称为主中断 (Primary Interrupter)，实际上 Interrupter 应该翻译为断路器，但是因本人喜好，这里和 Interrupt 混用，直接翻译为中断。剩余的所有中断都被称为次中断 (Secondary Interrupter)，一个控制器至少会支持一个中断，因此至少会有主中断，但可能没有次中断。
整个寄存器组的结构如下：

Offset	Size(Byte)	Name	Description
0x00	4	mgrReg	Interrupter Management Register，管理用寄存器。
0x04	4	mod	Interrupter Moderation，调节用寄存器，用于调节中断触发的频率。
0x08	4	eveSegTblSize	Event Ring Segment Table Size，事件环段的表格大小。
0x0c	4	Reserved and preserved	保留
0x10	8	eveSegTblAddr	Event Ring Segment Table Base Address，事件环段的表格基地址。
0x18	8	eveDeqPtr	Event Ring Dequeue Pointer，事件环出队指针。

mgrReg，中断的管理寄存器

Bit	Access	Description
0	R/WC	Interrupt Pending (IP)，表示是否有中断正在等待处理。在这一位被清空之前，不会有其他中断触发。
1	R/W	Interrupt Enable，是 (1) 否 (0) 激活中断。

Bit	Access	Description
2:31	R/W	Reserved and preserved

中断处理中，IP 位以及 usbStatus 的 EINT 的复位操作需要有一点讲究。这会在之后的算法和程序实现部分予以讲解。

mod，中断的调节

Bit	Access	Description
0:15	R/W	Interrupt Moderation Interval，中断的间隔最小值，单位为 $250ns$ 。
16:31	R/W	Interrupt Moderation Counter，倒计时，表示距离下一个中断的至少的时间。

实际的最小间隔和中断的最高频率计算如下：

$$\text{Interval} = \frac{1}{250 \times 10^{-9} \cdot (\text{Interrupt pre sec})}$$

$$\text{Interrupt pre sec} = \frac{1}{250 \times 10^{-9} \cdot \text{Interval}}$$

当 IP 位被复位之后，计时器就会根据 Interrupt Moderation Interval 进行倒计时。可以修改倒计时段来干扰倒计时过程。

eveSegTblSize，事件环段的表格大小

Bit	Access	Description
0:15	R/W	Event Ring Segment Table Size，大小
16:31	R/W	Reserved and preserved，保留

可行的最大大小由 hcsParams2 的 ERST Max 段指定。

eveSegTblAddr，事件环段的表格基地址

Bit	Access	Description
0:5	R/W	Reserved and preserved，保留
6:63	R/W	Event Ring Segment Table Base Address 的 Bit 6:63

实际上就是一个对 64 对齐的地址，但是写入的时候需要注意 Bit 0:5 的 preserved 。

eveDeqPtr，事件环出队指针

Bit	Access	Description
0:2	R/W	Dequeue ERST Segment Index，用于加速事件环已满状态 (Event Ring Full condition) 的检查，可以为 0
3	R/WC	Event Handler Busy，当前事件处理是否处于繁忙状态。当 IP 位被设置为 1 的时候，这一位被控制器设置为 1 。
4:63	R/W	Event Ring Dequeues Pointer (4:63) ，指针的 $4:63$ 位。

Doorbell Registers，Doorbell 寄存器组

似乎可以直接翻译为门铃寄存器组。但是无所谓了。使用 dbRegAddr = baseAddr + capRegs.dbOff 获取这个寄存器组的地址。这里包含若干个格式类似的，紧挨着的 Doorbell 寄存器，均为 32 位整数大小。
第一个寄存器为 Command Doorball Register ，接下来由 maxSlots 个 Device Slot Doorbell 寄存器，第 x 个（从 1 开始）写作 Device Slot $\#x$ Doorbell 。

Command Doorbell Register

Bit	Access	Description
0:7	R/W	Target，目标值，下面会提供每个值对应的含义。
8:31	R/W	Reserved and Zero'd ，保留

Deive Slot Doorbell，设备槽的 doorbell

Bit	Access	Description
0:7	R/W	Target，目标值，和 Command Doorbell Register 的 Target 段含义相同。
8:15	R/W	Reserved and Zero'd ，保留
16:31	R/W	Doorbell Stream ID，被使用 Streams 的端点 (endpoint) 使用，$0, 65534$ 和 65536 被保留，不能使用。

Target 值的解释

Target 的目标值不能为 0 该值被保留。
若为 1 ，则表示对控制端点 (Control endpoint) 进行入队指针更新 (Enqueue Pointer Update) 。
对于 $2, 2 \ldots, 31$ ，$\left\lfloor \frac{\text{Target}}{2} \right\rfloor$ 表示操作的端点，Target 为奇数则表示进行 IN 操作，为偶数则为 OUT 操作。
对于特定的 Target $= x$ ，表示对 $\lfloor \frac{x}{2} \rfloor$ 端点进行 IN（奇）/ OUT（偶）的进队指针更新。

对于 32 . . . , 247：被保留，不能使用。
对于 248 . . . , 255：对于不同的 Vendor ，有特定的含义。

需要操作系统申请的数据结构