Erledigt USB XHC ohne USBInjectAll.kext oder USB-Portlimit Patch zum Laufen bringen / Diskussionen

Beitrag von „apfelnico“ vom 12. Oktober 2020, 23:07

pgr69

Bei mir sehen die Ports so aus, in einer eigenen SSDT beschrieben:

Code

```
1. Scope (\_SB.PC00.XHCI.RHUB.HS01)
2. {
3. Name (_UPC, Package (0x04) // _UPC: USB Port Capabilities
4. {
5. 0xFF,
6. 0x03,
7. Zero,
8. Zero
9. })
10. }
```

Dabei steht in dem Package der erste Wert für den "Status" des Devices, lässt sich auch extra mit der Methode _STA festlegen. "0xFF" ist komplett an, "0x00" komplett aus, es gibt noch feinere Abstufungen dazwischen, die hier aber nicht benötigt werden.

Der zweite Parameter ist die Beschreibung um was für einen Port es sich handelt. "0x00" (beziehungsweise "Zero") bedeutet USB2, wogegen "0x03" USB3 festlegt (beides klassische Type A Buchse). Dann gibt es noch "0xFF" für einen internen USB2-Port (es gibt offenbar KEINEN "internen" USB3). "Intern" meint damit nicht unbedingt die Ports, die direkt auf einem Mainboard stecken. Denn diese können ja auch herausgeführt werden an "Gehäuseports". Diese sind somit je nach USB2 oder USB3 zu bezeichnen. Vielmehr bedeutet "Intern" zum einen für direkt auf dem Mainboard verdrahtete Geräte (herstellereigenes Bluetooth, RGB-Lichtsteuergedöhns etc), oder auch selbst eingesetzte interne Bluetoothkarte mit USB2-Uplink.

Wichtig: auch die USB2-Ports innerhalb eines USB3-Ports (der ja immer aus diesen zwei Komponenten besteht), wird als USB3, also "0x03" deklariert. Lediglich einzelne exklusive

USB2-Ports als "0x00". Dann gibt es noch USB-C (USB3.1/USB3.2). Hier gibt es wiederum zwei Varianten: "0x9" für diejenigen Ports, an denen man einen USB-C Gerät egal wie rum anstecken kann (der Stecker ist "Verdrehsicher", heißt er kann um 180º ebenfalls eingesteckt werden). Eine weitere Variante erlaubt letztendlich für den Nutzer das gleiche Verhalten, nur wird intern je nach Steckerdrehung ein zweiter eigener Port genutzt – in dem Fall ist es "0x0A". Letzteres findet man oft bei den "Type-E" Buchsen auf dem Mainboard (für Gehäuseports USB-C). Erkennen kann man das wunderbar mittels IORegistryExplorer. Auch hier gilt, die zugehörigen USB2-Ports – sofern vorhanden – werden genau so deklariert.

Dann gibt es noch ein exotisches Sonderformat von USB-C, welches man aber nicht unbedingt am Computer findet: USB2 exklusiv über USB-C. Dieses wird dann als "0x08" deklariert.

Die unteren beiden "Zero" haben sich mir auch noch nicht erschlossen.

Warum gibt es überhaupt nur 15 Ports (je Controller) bei Apple? Nun, direkt in Apples Geräte sind ja wesentlich weniger eingebaut, also ist "15" als Obergrenze mehr als ausreichend gewesen. Warum aber nun gerade "15", und nicht etwa "37"? 🙂

Die Ports werden, unabhängig von ihrer tatsächlichen Adresse, von "1 bis maximal 15" in macOS durchnummeriert. Also von "0x01" bis max. 0x0F". HEXADEZIMAL halt, sind eben 15 ...