Probing devices in other D states than 0

Introduction

In some cases it may be preferred to leave certain devices powered off for the entire system bootup if powering on these devices has adverse side effects, beyond just powering on the said device.

How it works

The _DSC (Device State for Configuration) object that evaluates to an integer may be used to tell Linux the highest allowed D state for a device during probe. The support for _DSC requires support from the kernel bus type if the bus driver normally sets the device in D0 state for probe.

The downside of using _DSC is that as the device is not powered on, even if there's a problem with the device, the driver likely probes just fine but the first user will find out the device doesn't work, instead of a failure at probe time. This feature should thus be used sparingly.

IÂ²C ---

If an IÂ²C driver indicates its support for this by setting the I2C_DRV_ACPI_WAIVE_D0_PROBE flag in struct i2c_driver.flags field and the _DSC object evaluates to integer higher than the D state of the device, the device will not be powered on (put in D0 state) for probe.

D states

The D states and thus also the allowed values for _DSC are listed below. Refer to [1] for more information on device power states.

```
Number  State   Description
0       D0      Device fully powered on
1       D1
2       D2
3       D3hot
4       D3cold  Off
```

References

[1] https://uefi.org/specifications/ACPI/6.4/02_Definition_of_Terms/Definition_of_Terms.html#device-power-state-definitions

Example

An ASL example describing an ACPI device using _DSC object to tell Operating System the device should remain powered off during probe looks like this. Some objects not relevant from the example point of view have been omitted.

```
Device (CAM0)
{
        Name (_HID, "SONY319A")
        Name (_UID, Zero)
        Name (_CRS, ResourceTemplate ()
        {
                I2cSerialBus(0x0020, ControllerInitiated, 0x00061A80,
                             AddressingMode7Bit, "\\_SB.PCI0.I2C0",
                             0x00, ResourceConsumer)
        })
        Method (_DSC, 0, NotSerialized)
        {
                Return (0x4)
        }
}
```