ALSD _ALI returns 0 (SMCLightSensor)

Beitrag von „BlvckBytes“ vom 15. Dezember 2021, 21:26

Hey!

Ich weis, das ist eigentlich bloß eine Spielerei... aber ich hätte echt gerne einen funktionierenden Ambient Light Sensor, nachdem dieser auf Windoof 10 auch sehr gute Dienste leistet. Mein System ist das Dell XPS 9700 mit i7-10875H und 4K-Display. Hab' mal den aktuellen EFI-folder angehangen.

Um mal kurz meinen bisherigen Ansatz zu beschreiben, zitiere ich das zu patchende Gerät aus meinen ACPI tables:

Code

```
1. Scope (_SB.PCI0.LPCB)
2. {
3. Device (ALSD)
4. {
5. Name (_HID, "ACPI0008" /* Ambient Light Sensor Device */) // _HID: Hardware ID
6. Method (_STA, 0, NotSerialized) // _STA: Status
7. {
8. If ((ALSE == 0x02))
9. {
10. Return (0x0B)
11. }
12.
13. Return (Zero)
14. }
15.
16. Method (_ALI, 0, NotSerialized) // _ALI: Ambient Light Illuminance
17. {
18. Return (((LHIH << 0x08) | LLOW))
19. }
20.
```

```
21. Name (_ALR, Package (0x05) // _ALR: Ambient Light Response
22. {
23. Package (0x02)
24. {
25. 0x46,
26. Zero
27. },
28.
29. Package (0x02)
30. {
31. 0x49,
32. 0x0A
33. },
34.
35. Package (0x02)
36. {
37. 0x55,
38. 0x50
39. },
40.
41. Package (0x02)
42. {
43. 0x64,
44. 0x012C
45. },
46.
47. Package (0x02)
48. {
49. 0x96,
50. 0x03E8
51. }
52. })
53. }
54. }
```

Alles anzeigen

Damit das Gerät in der IOReg aufscheint, musste ich natürlich ALSE auf 0x02 setzen, mittels SSDT. Gesagt getan, und schon mounted auch der LMU-Controller von Apple sowie der Eintrag in den Bildschirmeinstellungen (siehe Anhänge). Leider bewegt sich der Slider nicht, weshalb ich kurzerhand das Repo "VirtualSMC" geklont habe, um weitere Debugs einzufügen. Und zwar

wollte ich wissen, was _ALI zurückgibt, nachdem diese Schnittstelle von SMCLightManager gepollt wird.

Code

```
1. bool SMCLightSensor::refreshSensor(bool post) {
2. uint32_t lux = 0;
3. auto ret = alsdDevice->evaluateInteger("_ALI", &lux);
4. if (ret != kIOReturnSuccess)
5. lux = 0xFFFFFFFF; // ACPI invalid
6.
7. DBGLOG("alsd", "Lux currently is: %llu", lux);
8.
9. atomic_store_explicit(&currentLux, lux, memory_order_release);
10.
11. if (post) {
12. VirtualSMCAPI::postInterrupt(SmcEventALSChange);
13. poller->setTimeoutMS(SensorUpdateTimeoutMS);
14. }
15.
16. return ret == kIOReturnSuccess;
17. }
```

Alles anzeigen

Danach konnte ich mittels dmesg den Wert auslesen, welcher recht enttäuschend ausfällt:

Code

```
1. [ 791.886212]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
2. [ 792.887578]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
3. [ 793.888480]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
4. [ 794.890898]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
5. [ 795.893362]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
6. [ 796.895817]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
7. [ 797.896634]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
8. [ 798.899057]: SMCLightSensor alsd: @ (DBG) Lux currently is: 0
```

Nun machte ich mich auf die Suche nach LHIH und LLOW, nachdem diese zwei Variablen den Output bestimmen.

Code

```
1. blvckbytes@BlvckBook origin % grep -E 'LHIH|LLOW' *.dsl
2. DSDT.dsl: External (LHIH, UnknownObj) // (from opcode)
3. DSDT.dsl: External (LLOW, UnknownObj) // (from opcode)
4. DSDT.dsl: Return (((LHIH << 0x08) | LLOW))
5. SSDT-2-SaSsdt.dsl: LLOW, 8,
6. SSDT-2-SaSsdt.dsl: LHIH, 8,
```

Die Variablen befinden sich in einer OperationRegion auf dem SystemMemory, im globalen Scope. Ansonsten gibts wohl keine weiteren Zugriffe mehr darauf...

Code

```
1. OperationRegion (SANV, SystemMemory, 0x55DA3018, 0x01F4)
2. Field (SANV, AnyAcc, Lock, Preserve)
3. {
4. <omitted>
5. }
```

Und nun steh ich an. Die Werte werden wohl von einer ganz anderen Stelle im System über ihre bekannten Speicheradressen beschrieben (oder auch nicht, wie die 0 zeigt...). Hat jemand von Euch Ideen, wie ich weitermachen könnte?

Beitrag von „5T33Z0“ vom 15. Dezember 2021, 22:14

Anleitung hier: Ambient_Light_Sensor_(SSDT-ALS0/ALSD))

Beitrag von „BlvckBytes“ vom 15. Dezember 2021, 22:55

5T33Z0

Soweit mal nichts neues dort gefunden, außer:

> Zitat
>
> If there is an ambient light sensor device in the original ACPI and you want to force it to be enabled by the preset variable method, you need to pay attention to the existence of _SB.INI in the original ACPI. If it exists, please use method 2 to impersonate ALS0.

Und ja, diese Methode existiert in der Tat:

Code

```
1. Scope (_SB)
2. {
3. Method (_INI, 0, NotSerialized) // _INI: Initialize
4. {
5. Local0 = Zero
6. If ((DSCH != 0x44414548))
7. {
8. Local0 = One
9. }
10.
11. If ((DSCT != 0x4C494154))
12. {
13. Local0 = One
14. }
15.
16. If ((DSFH != 0x44414548))
17. {
18. Local0 = One
19. }
```

```
20.
21. If ((DSFT != 0x4C494154))
22. {
23. Local0 = One
24. }
25.
26. If ((DSPH != 0x44414548))
27. {
28. Local0 = One
29. }
30.
31. If ((DSPT != 0x4C494154))
32. {
33. Local0 = One
34. }
35.
36. If ((Local0 == One))
37. {
38. Sleep (0x7530)
39. }
40. Else
41. {
42. }
43.
44. EV4 (One, Zero)
45. }
46. }
```

Alles anzeigen

Verstehe den Zusammenhang aber noch nicht so ganz, weshalb ich jetzt ein fake ALS device erstellen soll, wenn doch ein natives vorhanden ist.

Beitrag von „5T33Z0“ vom 15. Dezember 2021, 23:34

Aber die SSDT-ALSD.dsl ist dir entgangen oder wie?

Abgesehen davon: Dein Controller ist längst mit dem Dienst AppleLMUController verknüpft!

Aktuelle version von SSDT-PNLF aus OpenCore Pakcage einbinden. SSDT-PNLF-CFL ist outdated.

Beitrag von „BlvckBytes“ vom 16. Dezember 2021, 02:44

5T33Z0

Danke dass Du mich zum Thema der PNLF aufmerksam gemacht hast!

Hab ich mal erneuert, Auswirkungen hatte es soweit keine, in Gesamtheit gesehen.

Bezüglich der SSDT-ALSD muss ich leider davon ausgehen, dass du meinen initialen Beitrag hier nicht vollständig durchgelesen hast, da ich diesen Patch schon längst ohne jegliche Dokumentationen implementiert habe - ist logisch und eigentlich Grundverständnis. Mein Problem liegt ja ganz wo anders. Klar, das Gerät registriert sich (Status 0xB), wodurch sich der LMUController daran mounted, aber das Ganze ist recht nutzlos, wenn die _ALI-Methode - wie in meinem Debug gezeigt - ständig einen Wert von 0 retourniert.

Mir kommt's so vor, als würde der Sensor einfach nicht in die nötigen Speicheradressen schreiben, weil er entweder A) nicht aktiviert oder B) nicht dazu aufgefordert wird.

Beitrag von „5T33Z0“ vom 17. Dezember 2021, 18:17

Damit der SMCLighSenor funktioniert muss ACPI0008/_ALI in der DSDT vorhanden sein.

Kannst ja ne SSDT schreiben, die den Wert da drun unter macOS einfach auf 1 setzt, mit If (_OSI ("Darwin")) usw.

Beitrag von „BlvckBytes“ vom 18. Dezember 2021, 04:36

5T33Z0

Verstehe ich das richtig: Dein Vorschlag ist es, den variablen Wert der Umgebungsbeleuchtungsstärke auf einen statischen bzw. festen Wert zu setzen? 

Beitrag von „5T33Z0“ vom 18. Dezember 2021, 17:09

Nein, verstehst Du falch. Es geht nicht darum, eine variable auf 1 zu setzen, sondern den Device unter macOS verfügbar zu machen, sodass 1. Der SMCLightSensor ihn findet und ihn 2. über die richtige SSDT-PNLF geregelt werden kann…

Dazu würde ich mit ner base config beginnen und nich mit einer mit 25 kexts und ACPI tables.

Ggf ACPI debugging einbinden.

Beitrag von „BlvckBytes“ vom 19. Dezember 2021, 19:15

5T33Z0

Verfügbar war er ja schon längst, bloß Werte wurden keine emittiert. Der Grund wieso ich das behaupten kann ist ja, weil bereits ACPI debugging betrieben wurde.

Ich sehe aktuell keine weiteren Lösungen für das Problem und melde mich daher eventuell später mal wieder, wenns was neues gibt. Sieht generell so aus, als würde zu diesem Nischenthema nicht sonderlich viel Wissen bzw. Erfahrung im Hackintoshbereich bestehen.

Beitrag von „5T33Z0“ vom 19. Dezember 2021, 19:20

Probiers mal damit: SSDT-ACPI0008-fix.dsl