Es tut sich wieder was bei Broadcom WLAN

Beitrag von „DerBeste“ vom 11. Dezember 2025, 03:03

> Zitat von mhaeuser
>
> MacGrummel Selbstverständlich können neue Treiber geladen werden. WiFi ist nur keine unterstützte Geräteklasse - jeder Mac hat bereits WiFi, also wird in die Richtung keine Arbeit gesteckt. Inwieweit es “unmöglich” ist im Vergleich zu den alten Ethernet-Emulatoren, weiß ich nicht - wenn es an mach_types.h scheitert, muss man wohl auf das nächste tolle LLM warten.

Okay, lass mich eines klarstellen, weil die Diskussion sich im Moment etwas im Kreis dreht: Das Problem ist nicht einfach nur an „mach/mach_types.h“ das ist lediglich ein Symptom, nicht die Ursache.

Als Entwickler solltest du das leicht reproduzieren können. Es geht nicht um eine einzelne Datei, sondern darum, wie DriverKit aufgebaut ist.

Im Grunde ist es so: DriverKit-Treiber dürfen nur die Header aus DriverKit und USBDriverKit verwenden, die dafür gedacht sind. Aber sobald du anfängst, echte Treiberlogik zu schreiben, wie zum Beispiel für USB, PCI oder Netzwerk nahe Funktionalität dann stößt du zwangsläufig auf Abhängigkeiten, die früher zulässige Kernel-APIs verwendet haben aber heute vollständig isoliert sind.

Das beinhaltet unter anderem:

* mach_*-Typen etc.

* IOUSB/IOService-Strukturen, die in DriverKit nicht übernommen wurden

* Synchronisations- und Memory-Primitive, für die es keinen Ersatz gibt

* Includes, die Xcode selbst über Standard-C++ Includes einzieht, auf die man selbst keinen Einfluss hat

Selbst wenn du jede direkte Referenz sauber entfernst, kommst du nicht weiter: Du ersetzt etwas Verbotenes, brauchst dafür etwas anderes, das auch nicht erlaubt ist, und dann geht der Build woanders schief.

Das ist kein Anfängerfehler und kein „falscher Include“.

Das ist ein unauflösbarer Zyklus. Du kannst es dir so vorstellen: Eine Katze, die ihren eigenen Schwanz hinterher jagt und egal, wie schnell sie hinterher läuft, wird sie niemals zum Ziel gelangen, weil das System bewusst so konstruiert wurde.

Ein weiterer Punkt, der gern übersehen wird:

Ein sauberer Build außerhalb von Xcode, etwa über ein Makefile oder CMake, ist faktisch unmöglich, da der Compiler während der Prüfung der erlaubten Includes scheitert, bevor man überhaupt funktionalen Code erreicht. Das ist kein Zufall, sondern Toolchain-Design.

Zu deiner Aussage „WiFi ist keine unterstützte Geräteklasse“: Genau das ist der Punkt. DriverKit suggeriert Offenheit, bietet sie aber real nur dort, wo Apple ohnehin vollständige Kontrolle besitzt. Dass alle Macs bereits WLAN haben, ist kein technisches Argument, sondern ein strategisches und erklärt exakt, warum diese Klassen bewusst ausgespart wurden.

Kurz gesagt: Ja, man kann formal neue DriverKit-Treiber laden. Aber in der Praxis sind sie so eingeschränkt, dass echte Hardware-Unterstützung nicht möglich ist.

Aber wenn du das nicht glauben magst, dann kannst du es gern selbst probieren:

* Einen eigenen USB- oder Netzwerk-Treiber schreiben

* Ohne private Frameworks

* Ohne alte Kext-Header

* Nur mit DriverKit

Das Ergebnis ist reproduzierbar und unabhängig vom Skill-Level. Das ist kein Versagen einzelner Entwickler.

Das System ist von Anfang an so gebaut, dass man nicht gewinnen kann.

 Ach, sorry, mit Hilfe der KI verfasst