# macOS Sierra 10.12.4 mit i7-5960X auf GA-X99-UD4

### Beitrag von „fblaese“ vom 20. April 2017, 13:51

Lange haben wir am X99 Hackintosh jetzt gebastelt, um Sierra 10.12.4 zum laufen zu bringen. Um euch möglicherweise ein bisschen Zeit zu sparen, möchte ich meine Erfahrungen mit einem Build auf X99 Basis mit i7-5960X hier teilen.

## Was geht:

- Bisher sehr stabil laufendes OS
- Audio
- LAN
- alle USB 2/3 Ports
- Basic Power Management

## Was NICHT geht:

- Standby (Tatsächlich scheint das aber nur an Kleinigkeiten zu liegen, unter Umständen lässt sich das lösen.)

## Was ich NICHT getestet habe:

- iMessage
- iTunes DRM Filme
- Wake on LAN

## Hardware:

GA-X99-UD4 (Rev 1.1)
i7-5960X
4x 8GB DDR4 RAM
GeForce Titan X

**Diese Hardware ist definitiv nichts für Hackintosh-Anfänger.**
**Daher hier auch keine Schritt für Schritt Anleitung, sondern nur eine Zusammenfassung der für diese Konfiguration spezifischen Probleme!**

**Gleich vorne weg**: die Titan X verändert nicht viel, daher wird sie im Text nicht mehr weiter erwähnt. Zum Booten ohne Webdriver das Bootflag `nv_disable=1` verwenden (Installer & System), im fertigen System den Webdriver installieren und danach das Bootflag wieder weglassen.

## 1. macOS bzw. Installer zum Booten bringen

Das größte Problem ist hier die CPU. Da das Power Control Device der 5960X ein BaseAddressRegister größer 1GB meldet, die IOPCIFamily.kext aber als Obergrenze (MAX_BAR_SIZE) 1GB festgelegt hat, hängt das System bei der PCI Configuration in einer Endlosschleife fest. (genaueres dazu hier)

Der gute Brumbaer ist diesem Problem nachgegangen und hat dafür einen Patch entwickelt, der von Clover on-thy-fly eingefügt werden kann. Dazu ist **für Sierra** folgendes in die config.plist einzufügen:

Code

```
<dict>
<key>Comment</key>
<string>5960X 10.12</string>
<key>Disabled</key>
<false/>
<key>Find</key>
<data>
SIH7AAAAQA==
</data>
<key>Name</key>
<string>IOPCIFamily</string>
<key>Replace</key>
<data>
SIH7AAAAgA==
</data>
```

16. </dict>

Alles anzeigen

Außerdem scheint es mit dem X99 Chipsatz Probleme mit dem OsxAptioFix zu geben. Genaueres dazu gibt es hier.
Anstatt der gewöhnlichen Fixes den **OsxAptioFix2Drv-free2000.efi** unter `/EFI/CLOVER/drivers64UEFI` einfügen, damit sollte das Problem beseitigt sein.

Außerdem waren das Bootflag `npci=0x2000` und natürlich die **FakeSMC.kext** notwendig.
Es wird für diese Konfiguration an vielen Stellen geschrieben, dass der **VoodooTSCSync.kext** notwendig ist. Das war bei diesem System bisher **nicht** notwendig.

Soweit sollte der Installer ausgehend von der Default Config von Clover auch schon starten.

## 2. USB 3.0 Ports

Ohne weitere Patches erkennt macOS alle USB-Ports nur als USB 2 Ports.
Dafür ist zunächst der **USBInjectAll.kext** von RehabMan inklusive dem im Readme erwähten **XHCI-x99-injector.kext** notwendig, damit die XHCI Controller ebenfalls erkannt werden. Weiterhin gibt es in macOS ein 15-port Limit für die XHCI Controller. Das lässt sich mit folgendem Patch lösen, der das Port Limit auf 20 anhebt und somit alle Ports nutzbar macht (ebenfalls im Readme vom USBInjectAll.kext):

Code

1. <dict>
2. <key>Comment</key>
3. <string>increase xhci port limit</string>
4. <key>Disabled</key>
5. <false/>
6. <key>Find</key>
7. <data>
8. g710////EA==
9. </data>

10. <key>Name</key>
11. <string>AppleUSBXHCIPCI</string>
12. <key>Replace</key>
13. <data>
14. g710////Fg==
15. </data>
16. </dict>

Alles anzeigen

## 3. Audio, LAN, SMBIOS und Feinheiten

Auf dem GA-X99-UD4 befindet sich ein ganz gewöhnlicher **ALC-11505**, für diesen ist aktuell die **AppleALC.kext** mit **Layout-ID 1** die beste Wahl. Dazu gibts bereits Anleitungen ohne Ende, daher gehe ich hier nicht näher darauf ein.
**ACHTUNG:** Für dieses Mainboard wird an manchen stellen die VoodooHDA empfohlen, was allerdings eher schlecht funktioniert und bei diesem Chip keinen Sinn macht.

Auf dem Board befindet sich außerdem ein Intel NIC, des sich mit der **AppleInstelE1000e.kext** zum Laufen bringen lässt. Auch diese wurde im Forum bereits ausreichend behandelt.

Als SMBIOS haben wir für dieses System den **MacPro5,1** verwendet, das funktioniert bisher super. Dafür ist ein weiterer Patch notwendig um das Laden der AppleTyMCE Treiber für ECC Speicher zu verhindern (Quelle

Code

1. <dict>
2. <key>Name</key>
3. <string>AppleTyMCEDriver</string>
4. <key>Find</key>
5. <data>cgoATWFjUHJvNCwxAE1hY1BybzUsMQBY</data>
6. <key>Replace</key>
7. <data>cgoAAAAAAAAAAAAAAAAAAAAAAAAAABY</data>
8. </dict>

Schlussendlich noch etwas um ein wenig Strom zu sparen: Grundlegendes Speed Stepping.
macOS kann mit dieser Konfiguration nicht umgehen, daher wird dem System per FakeCPUID eine andere CPU untergeschoben. Das macht außerdem die **NullCPUPowerManagement.kext** notwendig.
In Clover lässt sich das mit folgendem Code unter KernelAndKextPatches machen:

Code

1. <key>FakeCPUID</key>
2. <string>0x0306a0</string>

**Ich habe gelesen, dass es dafür eine bessere Lösung gibt, hatte aber bisher keine Zeit, das nochmal genauer zu untersuchen.**

## 4. Wissenswertes

Alle hier erwähnten Kernel Extensions lassen sich von Clover on-the-fly patchen, es ist **KEINE Modifikation** von `/System/Library/Extensions` oder irgendeiner anderen Stelle der Systempartition notwendig, was in meinen Augen eine sehr saubere Lösung ist.

## 5. Tl;dr

> Zitat
>
> **macOS Version**: Sierra 10.12.4
> **Bootloader**: Clover als UEFI Installation mit Default Config
> **SMBIOS**: MacPro5,1
> **Patches**: MacPro5,1 Patch, 5960X Patch by Brumbaer, XHCI Port Limit Patch
> **Bootflags**: npci=0x2000
> **Kexts**: FakeSMC, AppleALC, AppleIntelE1000e, NullCPUPowerManagement, USBInjectAll

So, das sollte es soweit gewesen sein, ich hoffe ich habe nichts vergessen.
Bei Problemen gerne im Forum oder unter diesem Thread fragen.

An dieser Stelle neben einem Dank an die üblichen Verdächtigen (e.g. Clover, AppleALC usw.) nochmal ein spezieller Dank an:
- **Brumbaer** für den 5960X Patch
- **RehabMan** für den x99 USB Patch
- **Nick Woodhams** für den X99 OsxAptioFix2

Ohne euch wäre diese Konfiguration nicht möglich gewesen!

---

## Beitrag von „al6042“ vom 20. April 2017, 14:00

Wow,
vielen Dank für den tollen Beitrag. 👍

Zum Sleep:
Hast du eine SSDT per ssdtprgen.sh erstellen lassen?
Wenn ja, sollte der NullCPUPowerManagement.kext eigentlich nicht mehr nötig sein und der Sleep-Mode auch funktionieren, wenn ich mich nicht irre...

---

## Beitrag von „Robin0815“ vom 24. April 2017, 12:36

Ich hab das gleiche Problem mit dem Standby auf dem X99-DELUXE II .. da hilft irgendwie auch keine SDDT. Zumindest habe ich es nicht hinbekommen. Ich bin mit meinem Build mittlerweile soweit, dass xpcm auch läuft, werde meinen Guide die Tage dann auch mal zusammenstellen.

Soweit ich das nachfollziehen konnte sind die NullCPUPowerManagement/AppleIntelCPU.. kexts gar nicht mehr notwendig. Zumindest läuft meiner ohne und ich hab keine Probleme. Benchmark ist aktuell bei stabilen 24.000, hab aber das Gefühl da geht mehr.

---

## Beitrag von „griven“ vom 26. April 2017, 21:29

Das hast Du ganz richtig nachvollzogen 
Die NullCPUPowerManagement.kext verhindert im Grunde nur das die AppleIntelCPUPowerManagement.kext geladen da aber dank XCPM die AppleIntelCPUPowerManagement.kext gar nicht erst geladen wird ist auch die NullCPUPowerManagement.kext jetzt nutzlos geworden. XCPM ist im Grunde der Nachfolger des AICPM und verlagert das PowerManagement direkt in den Kernel.