

POPULAR

ALL

RANDOM

USERS

ASKREDDIT

WORLDNEWS

VIDEOS

FUNNY

TODAYILEARNED

PICS

GAMING

MOVIES

NEWS

GIFS

Want to join? Log in or sign up in seconds.

AMD

Enabling today.
Inspiring tomorrow.

COMMENTS

OTHER DISCUSSIONS (2)

Filters: AMD CPU GPU NEWS REVIEW RUMOR PHOTO

112

Request

Threadripper KVM GPU Passthru: Testers needed

self.Amd

Submitted 5 months ago * by HyenaCheeseHeads x2

For a while now it has been apparent that PCI GPU passthrough using VFIO-PCI and KVM on Threadripper is a bit broken.

This manifests itself in a number of ways: When starting a VM with a passthru GPU it will either crash or run extremely slowly without the GPU ever actually working inside the VM. Also, once a VM has been booted the output of lspci on the host changes from one kind of output to another. Finally the output of dmesg suggests an issue bringing the GPU up from D0 to D3 power state.

An example of this lspci before and after VM start, as well as dmesg kernel buffer output is included here for the 7800GTX:

```
08:00.0 VGA compatible controller: NVIDIA Corpo

[ 121.409329] virbr0: port 1(vnet0) entered bl
[ 121.409331] virbr0: port 1(vnet0) entered di
[ 121.409506] device vnet0 entered promiscuous
[ 121.409872] virbr0: port 1(vnet0) entered bl
[ 121.409874] virbr0: port 1(vnet0) entered li
[ 122.522782] vfio-pci 0000:08:00.0: enabling
[ 123.613290] virbr0: port 1(vnet0) entered le
[ 123.795760] vfio_bar_restore: 0000:08:00.0 r
...
[ 129.534332] vfio-pci 0000:08:00.0: Refused t

08:00.0 VGA compatible controller [0300]: NVIDI
!!! Unknown header type 7f
Kernel driver in use: vfio-pci
```

Notice that lspci reports revision FF and can no longer read the header type correctly. Testing revealed that pretty much all graphics cards except Vega would exhibit this behavior, and indeed the output is very similar to the above.

Reddit user /u/wendelltron and others suggested that the D0->D3 transition was to blame. After having gone through a brute-force exhaustive search of the BIOS, kernel and vfio-pci settings for power state transitions it is safe to assume that this is probably not the case since none of it helped.

AMD representative /u/AMD_Robert suggested that only GPUs with EFI-compatible BIOS should be able to be used for passthru in an EFI environment, however, testing with a modern 1080GTX with EFI bios support failed in a similar way:

POWERING PRODUCTS YOU LOVE.

AMD

118,868 READERS

Subscribe

search

SUBMIT LINK

SUBMIT TEXT

This post was submitted on 30 Nov 2017

112 points (99% upvoted)

shortlink: <https://redd.it/7gp1z7>

username

password

☐ remember me
 [reset password](#)

Login

53% OF DATA BREACHES INITIALLY TARGET APPS

READ OUR eGUIDE →

WE MAKE APPS

FASTER. SMARTER. SAFER.

AMD

2,841 CURRENTLY ONLINE

IrAMD Community

DISCORD

AMD Official Community

DISCORD

Latest Drivers & Tech Support

Radeon Software Adrenalin Edition 18.3.4 - March 26, 2018

Display your AMD Adrenalin performance logs with Adrenalin Charts

```
42:00.0 VGA compatible controller: NVIDIA Corpo
and then
42:00.0 VGA compatible controller: NVIDIA Corpo
!!! Unknown header type 7f
```

Common to all the cards was that they would be unavailable in any way until the host system had been restarted. Any attempt at reading any register or configuration from the card would result in all-1 bits (or FF bytes). The bitmask used for the headers may in fact be what is causing the 7f header type (and not an actual header being read from the card). Not even physically unplugging and re-plugging the card, rescanning the PCIe bus (with `/sys/bus/pci/rescan`) would trigger any hotplug events or update the card info. Similarly, starting the system without the card and plugging it in would not be reflected in the PCIe bus enumeration. Some cards, once crashed, would show spurious PCIe ACS/AER errors, suggesting an issue with the PCIe controller and/or the card itself. Furthermore, the host OS would be unable to properly shut down or reboot as the kernel would hang when everything else was shut down.

A complete dissection of the vfio-pci kernel module allowed further insight into the issue. Stepping through VM initialization one line at a time (yes this took a while) it became clear that the D3 power issue may be a product of the FF register issue and that the actual instruction that kills the card may have happened earlier in the process. Specifically, the function `drivers/vfio/pci/vfio_pci.c:vfio_pci_ioctl`, which handles requests from userspace, has entries for `VFIO_DEVICE_GET_PCI_HOT_RESET_INFO` and `VFIO_DEVICE_PCI_HOT_RESET` and the following line of code is exactly where the cards go from active to "disconnected" states:

```
if (!ret)
    /* User has access, do the reset */
    ret = slot ? pci_try_reset_slot(vdev->slot,
                                   pci_try_reset_bus(vdev->pdev->bus));
```

Commenting out this line allows the VM to boot and the GPU driver to install. Unfortunately for the nVidia cards my testing stopped here as the driver would report the well known error 43/48 for which they should be ashamed and **shunned** by the community. For AMD cards a R9 270 was acquired for further testing.

The reason this line is in vfio-pci is because VMs do not like getting an already initialized GPU during boot. This is a well-known problem with a number of other solutions available. By disabling the line it is necessary to use one of the other solutions when restarting a VM. For Windows you can disable the device in Device Manager before reboot/shutdown and re-enable it again after the restart - or use login/logoff scripts to have the OS do it automatically.

Unfortunately another issue surfaced which made it clear that the VMs could only be stopped once even though they could now be rebooted many times. Once they were shut down the cards would again go into the all FF "disconnect" state. Further dissection of vfio-pci revealed another instance where an attempt to reset the slot that the GPU is in was made: in `drivers/vfio/pci/vfio_pci.c:vfio_pci_try_bus_reset`

```
if (needs_reset)
    ret = slot ? pci_try_reset_slot(vdev->slot,
                                   pci_try_reset_bus(vdev->pdev->bus));
```

- [Submit Driver Feedback to AMD](#)
- [April Tech Support Megathread](#)

General Information

Welcome to [/r/AMD](#)! In this subreddit, we discuss and share news, rumors, ideas, and knowledge relating to AMD, their hardware and software products, and the silicon industry.

Please note that this subreddit is community run and does not represent AMD unless otherwise specified.

Rules

Rule 1: Tech support questions are only allowed in [tech support megathreads](#) and must instead be posted at [/r/AMDHelp](#) or [/r/techsupport](#). Any other tech support posts will be removed at moderator discretion.

Rule 2: No referral links, including Amazon! Product links are fine, affiliate or referral links that benefit you are not.

Rule 3: Be civil and obey reddit etiquette. Please remember that behind every poster is a human. This means no brigade incitements, personal attacks, or "mentioning" a user in order to annoy or harass them, etc.

Rule 4: Use of slurs of any kind, racial, homophobic, or whatever, in any context will result in a ban. This includes derogatory comments such as "retard". There's no need for petty insults on this sub.

Rule 5: All posts must be related to AMD or AMD products. Example of okay: RX480 vs 1060. Not okay: GTX 1060 vs 1080.

Rule 6: No religion/politics! There are plenty of other places for that.

Rule 7: Use original sources. Copy-paste articles sourced from other websites are not allowed. Quotes are fine, but pasting the entire article in a textpost is not. Original articles are always better than a reddit textpost.

Rule 8: Shitposts, memes, and plain box pictures are not allowed as linkposts (you can still include them *within* normal posts or comments). Visit [/r/AyyMD](#) for dank shitposts and memes. Strawpolls are not allowed.

Rule 9: The moderators of [/r/AMD](#) reserve the right to allow posts or comments that could technically break rules #1 (tech support) or #8 (no memes), when a situation has arisen where the post is especially necessary, funny, educational, or useful to the users of the subreddit. Reports are *always* welcome, but remember that content sometimes remains up due to this rule, rather than because of lack of moderator work.

Rule 10: No bamboozling

Links

- Tech Support Megathreads:
[Apr'18](#) [Mar'18](#) [Feb'18](#) [Jan'18](#) [Dec'17](#) [Nov'17](#) [Oct'17](#) [Sep'17](#) [Aug'17](#) [Jul'17](#) [Jun'17](#) [May'17](#) [Apr'17](#) [Mar'17](#) [Feb'17](#) [Jan'17](#) [Dec'16](#) [Nov'16](#)
- [Crossfire Compatible Games](#)
- [How AMD's GlobalFoundries chips are made](#)
- [AMD website](#)
- [AMD Blog](#)

When this line is instead skipped, a VM that has had its GPU properly disabled via Device Manager and has been properly shutdown is able to be re-launched or have another VM using the same GPU launched and works as expected.

I do not understand the underlying cause of the actual issue but the workaround seems to work with no issues except the annoyance of having to disable/re-enable the GPU from within the guest (like in ye olde days). Only speculation can be given to the real reason of this fault; the hot-reset info gathered by the ioctl may be wrong, but the ACS/AER errors suggest that the issue may be deeper in the system - perhaps the PCIe controller does not properly re-initialize the link after hot-reset just as it (or the kernel?) doesn't seem to detect hot-plug events properly even though acpihp supposedly should do that in this setup.

Here is a "screenshot" of Windows 10 running the Unigine Valley benchmark inside a VM with a Linux Mint host using KVM on Threadripper 1950x and an R9 270 passed through on an Asrock X399 Taichi with 1080GTX as host GPU:

<https://imgur.com/a/0HggN>

This is the culmination of many weeks of debugging. It is interesting to hear if anyone else is able to reproduce the workaround and can confirm the results. If more people can confirm this then we are one step closer to fixing the actual issue.

If you are interested in buying me a pizza, you can do so by throwing some Bitcoin in this direction:
1KToxJns2ohhX7AMTRrNtvzZJsRtwvsppx

Also, English is not my native language so feel free to ask if something was unclear or did not make any sense.

Update 1 - 2017-12-05:

Expanded search to non-gpu cards and deeper into the system. Taking memory snapshots of pcie bus for each step and comparing to expected values. Seem to have found something that may be the root cause of the issue. Working on getting documentation and creating a test to see if this is indeed the main problem and to figure out if it is a "feature" or a bug. Not allowing myself to be optimistic yet but it looks interesting, it looks fixable at multiple levels.

Update 2 - 2017-12-07:

Getting a bit closer to the real issue. The issue seems to be that KVM performs a bus reset on the secondary side of the pcie bridge above the GPU being passed through. When this happens there is an unintended side effect that the bridge changes its state somehow. It does not return in a useful configuration as you would expect and any attempt to access the GPU below it results in errors.

Manually storing the bridge 4k configuration space before the bus reset and restoring it immediately after the bus reset seems to magically bring the bridge into the expected configuration and passthru works.

The issue could probably be fixed in firmware but I'm trying to find out what part of the configuration space is fixing the issue and causing the bridge to start working again. With that information it will be possible to write a targeted patch for this quirk.

Update 3 - 2017-12-10:

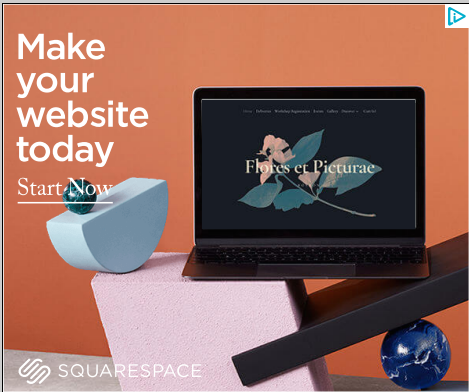
Begun further isolation of what particular registers in the config space are affected unintentionally by the secondary bus reset on the bridge. This is difficult work because the changes are seemingly invisible to the kernel, they happen only in the hardware.

So far at least registers 0x19 (secondary bus number) and 0x1a (subordinate bus number) are out of sync with the values in the config space. When a bridge is in faulty mode, writing their

- [AMD Twitch Channel](#)
- [@AMDGaming on Twitter](#)

Related Subreddits

- [/r/hardware](#)
(for general discussion of hardware)
- [/r/Intel](#)
(The main CPU competitor)
- [/r/NVIDIA](#)
(The only GPU competitor)
- [/r/PCGaming](#)
- [/r/AskPCGamers](#)
- [/r/AskLinuxUsers](#)
- [/r/CIRT](#)
(Can I run this?)
- [/r/PCMasterRace](#)
(For all things PC)
- [/r/ultrawidemasterrace](#) (Where wider is better)
- [/r/AyyMD](#)
(ADVANCED MEME DEVICES)
- [/r/intelmao](#) (The Shintel wannabe version of AyyMD)
- [/r/AMDHelp](#)
(Best place for tech support outside of the monthly megathread)
- [/r/techsupport](#)
- [/r/AMD_Stock](#) (Buy, hold, or sell?!)



MODERATORS

message the moderators

Tizaki

M7

RenegadeAI

M5

CSS

milocher

M5

BioGenx2b

M5

RTG

1700X + RX 480

bizude

M5

2200G

tugasdocri

M5

RTG Marketing

Isaac131

M3

R5 1600X / R9 290

PhoBoChai

M3

2600K + Vega 56

dayman56

M3

R7 1700 3.9 | GTX 970

AMD_Bot

codeboop

...and 1 more »

<

>

discussions in [r/Amd](#)

x

771 · 150 comments

Finally gone full AMD. Goodbye Novideo!

Theme based on [/r/Naut](#)

already existing value back to them brings the bridge back into working mode.

Update 4 - 2017-12-11 ("the ugly patch"):

After looking at the config space and trying to figure out what bytes to restore from before the reset and what bytes to set to something new it became clear that this would be very difficult without knowing more about the bridge.

Instead a different strategy was followed: Ask the bridge about its current config after reset and then set its current config to what it already is; byte by byte. This brings the config space and the bridge back in sync and everything, including reset/reboot/shutdown/relaunch without scripts inside the VM, now seems to work with the cards acquired for testing. Here is the ugly patch for the brave souls who want to help test it.

Please, if you already tested the workaround: revert your changes and confirm that the bug still exists before testing this new ugly patch:

In `/drivers/pci/pci.c`, replace the function `pci_reset_secondary_bus` with this alternate version that adds the ugly patch and two variables required for it to work:

```
void pci_reset_secondary_bus(struct pci_dev *dev)
{
    u16 ctrl;
    int i;
    u8 mem;

    pci_read_config_word(dev, PCI_BRIDGE_CONTROL, &ctrl);
    ctrl |= PCI_BRIDGE_CTL_BUS_RESET;
    pci_write_config_word(dev, PCI_BRIDGE_CONTROL, ctrl);
    /*
     * PCI spec v3.0 7.6.4.2 requires minimum T
     * this to 2ms to ensure that we meet the m
     */
    msleep(2);

    ctrl &= ~PCI_BRIDGE_CTL_BUS_RESET;
    pci_write_config_word(dev, PCI_BRIDGE_CONTROL, ctrl);

    // The ugly patch
    for (i = 0; i < 4096; i++){
        pci_read_config_byte(dev, i, &mem);
        pci_write_config_byte(dev, i, mem);
    }

    /*
     * Trhfa for conventional PCI is 2^25 clock
     * Assuming a minimum 33MHz clock this resu
     * delay before we can consider subordinate
     * be re-initialized. PCIe has some ways t
     * but we don't make use of them yet.
     */
    ssleep(1);
}
```

The idea is to confirm that this ugly patch works and then beautify it, have it accepted into the kernel and to also deliver technical details to AMD to have it fixed in BIOS firmware.

Update 5 - 2017-12-20:

Not dead yet!

Primarily working on communicating the issue to AMD. This is slowed by the holiday season setting in. Their feedback could potentially help make the patch a lot more acceptable and a lot less ugly.

Update 6 - 2018-01-03 ("the java hack"):

AMD has gone into some kind of ninja mode and has not provided any feedback on the issue yet.

Due to popular demand a userland fix that does not require recompiling the kernel was made. It is a small program that runs as any user with read/write access to sysfs (this small guide assumes "root"). The program monitors any PCIe device that is connected to VFIO-PCI when the program starts, if the device disconnects due to the issues described in this post then the program tries to re-connect the device by rewriting the bridge configuration.

This program pokes bytes into the PCIe bus. Run this at your own risk!

Guide on how to get the program:

- Go to <https://pastebin.com/iYg3Dngs> and hit "Download" (the MD5 sum is supposed to be 91914b021b890d778f4055bcc5f41002)
- Rename the downloaded file to "ZenBridgeBaconRecovery.java" and put it in a new folder somewhere
- Go to the folder in a terminal and type "javac ZenBridgeBaconRecovery.java", this should take a short while and then complete with no errors. You may need to install the Java 8 JDK to get the javac command (use your distribution's software manager)
- In the same folder type "sudo java ZenBridgeBaconRecovery"
- Make sure that the PCIe device that you intend to passthru is listed as monitored with a bridge
- Now start your VM

If you have any PCI devices using VFIO-PCI the program will output something along the lines of this:

```
-----
Zen PCIe-Bridge BAR/Config Recovery Tool, rev 1
-----
Wed Jan 03 21:40:30 CET 2018: Detecting VFIO-PC
Wed Jan 03 21:40:30 CET 2018:   Device: /sys/de
Wed Jan 03 21:40:30 CET 2018:       Bridge: /sy
Wed Jan 03 21:40:30 CET 2018:   Device: /sys/de
Wed Jan 03 21:40:30 CET 2018:       Bridge: /sy
Wed Jan 03 21:40:30 CET 2018:   Device: /sys/de
Wed Jan 03 21:40:30 CET 2018:       Bridge: /sy
Wed Jan 03 21:40:30 CET 2018:   Device: /sys/de
Wed Jan 03 21:40:30 CET 2018:       Bridge: /sy
Wed Jan 03 21:40:30 CET 2018: Monitoring 4 devi
```

And upon detecting a bridge failure it will look like this:

```
Wed Jan 03 21:40:40 CET 2018: Lost contact with
Wed Jan 03 21:40:40 CET 2018:   Recovering 512
Wed Jan 03 21:40:40 CET 2018:   Bridge config w
Wed Jan 03 21:40:40 CET 2018:   Recovered bridg
Wed Jan 03 21:40:40 CET 2018: Re-acquired conta
```

This is not a perfect solution but it is a stopgap measure that should allow people who do not like compiling kernels to experiment with passthru on Threadripper until AMD reacts in some way. Please report back your experience, I'll try to update the program if there are any issues with it.

116 comments [share](#) [save](#) [hide](#) [report](#)

all 116 comments

sorted by: best ▼

Want to add to the discussion?

Post a comment!

CREATE AN ACCOUNT

[-] pfbangs • AMD all day • 28 points 5 months ago

Thanks for doing the (exhaustive) legwork on this.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] Rhynri • 5 points 4 months ago

Yes, exactly as pfbangs mentioned. I built my TR build for the exclusive purpose of being a multihead machine that runs my entire household. This GPU issue has stubbornly stood in the way from day one. You are a credit to humanity, good sir, even if your head is cheesy.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] Urishima • 23 points 5 months ago

You should probably try to get in touch with gnif over at level1techs, he's working on that stuff as well. He's the guy who found the fix for the NPT bug. <https://forum.level1techs.com/t/a-little-teaser-of-what-is-to-come>

Unless you are gnif, then you don't need to try. You can touch yourself anytime you want.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] ct_the_man_doll • 7 points 5 months ago*

You should probably try to get in touch with gnif over at level1techs I posted a reply on the thread; hopefully he sees it.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] gnif2 • 9 points 5 months ago

Thanks! :D. Yes I saw it. Excellent work [/u/HyenaCheeseHeads](#)

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] gnif2 • 3 points 3 months ago*

I finally have a TR system that exhibits this behavior and I have spent the last week going through everything, [/u/HyenaCheeseHeads](#) has done excellent work as I can confirm he has nailed the problem down to the dummy host bridge.

You can rewrite the configuration space from user space with the following command (adjusting the ID as described below)

```
sudo dd if="/sys/bus/pci/devices/0000:40:03.1,
```

To obtain the correct bridge ID run

```
lspci -tv
```

Then look for your passed through device, in my case it is the GTX 1080Ti at "0000:42:00", it can be seen on the dummy hub "0000:40:03.1" in the below output.

```
--[0000:40]--00.0 Advanced Micro Devices,
|      +-00.2 Advanced Micro Devices,
|      +-01.0 Advanced Micro Devices,
|      +-01.1-[41]----00.0 Samsung Ele
|      +-02.0 Advanced Micro Devices,
|      +-03.0 Advanced Micro Devices,
|      +-03.1-[42]--+-00.0 NVIDIA Corp
|      |      \-00.1 NVIDIA Corp
|      +-04.0 Advanced Micro Devices,
|      +-07.0 Advanced Micro Devices,
|      +-07.1-[43]--+-00.0 Advanced Mi
|      |      +-00.2 Advanced Mi
```

```

|          |          \-00.3  Advanced Micro Devices, Inc.
|          +-08.0  Advanced Micro Devices, Inc.
|          \-08.1-[44]---+-00.0  Advanced Micro Devices, Inc.
|          |          \-00.2  Advanced Micro Devices, Inc.
\-[0000:00]---+-00.0  Advanced Micro Devices, Inc.
|          +-00.2  Advanced Micro Devices, Inc.
|          +-01.0  Advanced Micro Devices, Inc.
|          +-01.1-[01-06]---+-00.0  Advanced Micro Devices, Inc.
|          |          +-00.1  Advanced Micro Devices, Inc.
|          |          \-00.2-[02-06]---+-00.0  Advanced Micro Devices, Inc.
|          |          |          \-00.3  Advanced Micro Devices, Inc.
|          |          +-02.0  Advanced Micro Devices, Inc.
|          |          +-03.0  Advanced Micro Devices, Inc.
|          |          +-03.1-[07-09]---+-00.0-[08-09]---+-00.0  Advanced Micro Devices, Inc.
|          |          |          \-00.1  Advanced Micro Devices, Inc.
|          |          |          \-00.2  Advanced Micro Devices, Inc.
|          |          |          \-00.3  Advanced Micro Devices, Inc.
|          |          +-04.0  Advanced Micro Devices, Inc.
|          |          +-07.0  Advanced Micro Devices, Inc.
|          |          +-07.1-[0a]---+-00.0  Advanced Micro Devices, Inc.
|          |          |          +-00.2  Advanced Micro Devices, Inc.
|          |          |          \-00.3  Advanced Micro Devices, Inc.
|          |          +-08.0  Advanced Micro Devices, Inc.
|          |          +-08.1-[0b]---+-00.0  Advanced Micro Devices, Inc.
|          |          |          +-00.2  Advanced Micro Devices, Inc.
|          |          |          \-00.3  Advanced Micro Devices, Inc.
|          +-14.0  Advanced Micro Devices, Inc.
|          +-14.3  Advanced Micro Devices, Inc.
|          +-18.0  Advanced Micro Devices, Inc.
|          +-18.1  Advanced Micro Devices, Inc.
|          +-18.2  Advanced Micro Devices, Inc.
|          +-18.3  Advanced Micro Devices, Inc.
|          +-18.4  Advanced Micro Devices, Inc.
|          +-18.5  Advanced Micro Devices, Inc.
|          +-18.6  Advanced Micro Devices, Inc.
|          +-18.7  Advanced Micro Devices, Inc.
|          +-19.0  Advanced Micro Devices, Inc.
|          +-19.1  Advanced Micro Devices, Inc.
|          +-19.2  Advanced Micro Devices, Inc.
|          +-19.3  Advanced Micro Devices, Inc.
|          +-19.4  Advanced Micro Devices, Inc.
|          +-19.5  Advanced Micro Devices, Inc.
|          +-19.6  Advanced Micro Devices, Inc.
|          \-19.7  Advanced Micro Devices, Inc.

```

Update Further reading shows the ugly patch might not be that ugly.

According to the PCI bridge spec, when the secondary interface reset line is asserted any buffers must be reconfigured.

The bridge's secondary bus interface and any buffers between the two interfaces (primary and secondary) must be initialized back to their default state whenever this bit is set

The 'ugly fix' might actually be the correct fix.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [HyenaCheeseHeads\[S\]](#) • 3 points 3 months ago*

 TLDR: Yes.

Slightly longer comment:

We agree on both the issue and the solution - although adding a way to disable the patch from the kernel command line would probably be a good idea in the case of a potential incompatible bridge somewhere out there. The ugliness of the ugly patch is mostly related to how it

copies the configuration - it applies itself for all systems and it completely ignores sizes of the registers and the fact that not all registers really need to be copied. The patch you posted to patchwork is closer to what is likely needed in order to get it accepted. If you are up for shining it up a bit and drumming up a bit of discussion on it through LKML that would be awesome! - my primary interest in this has mostly been in the technical aspects of getting it to work in the first place anyways.

Even longer, very technical comment:

When viewing this from a hardware engineering perspective there is a fairly reasonable explanation as to why this hasn't been a bigger issue on other hardware yet (even though some other hardware has been showing the exact same pattern):

Normally, for simple hardware, the configuration registers are directly tied to the function that they are related to; there is no indirection. Writing a register either triggers an action or sets a bunch of flip-flops directly in a hardware module somewhere. The configuration register *IS* the hardware configuration.

The data fabric in the Zen core is *not normal*. This is not meant in a derogatory way, quite the opposite in fact. It is my understanding that the data fabric has more in common with an FPGA than a usual PCIe controller. The fabric binds the many 12Gbit general purpose PHYs (the parts that put electricity on the wires and pins) with the rest of the chip. It is extremely configurable and capable of creating many different PCIe configurations with differing widths and speeds. It can also create sata, ethernet or xGMI through those same PHYs. The actual configuration of a Zen-based chip is based on a set of fuses and run-time configuration uploaded to the control fabric by the motherboard BIOS through calls in AGESA. The motherboard vendors can configure it to match the physical properties of their PCIe ports on the board; and they can also allow stuff like splitting an x16 port into 4x4 ports for nvme riser cards etc.

This is where speculation begins: When a new port is configured a state machine is also allocated to it, and this state machine is partly based in firmware rather than just hardware. This state machine is responsible for implementing most of the PCIe spec and translating the actions from the high-bandwidth internal PCIe to the (equal or lower bandwidth) external PCIe through the available underlying hardware. This is the bridge that we are talking to. My guess is that it has a bit of memory allocated to it to keep track of the config registers and it has direct access to the configured PCIe hardware MACs which are connected to the PHYs through some muxing network of some sort. This means that the configuration registers are merely used as part of the state machine and the actual hardware state and configuration is controlled by the state machine too. In other words the configuration registers are indirectioned.

The indirection could very well be the source of the confusion surrounding the PCIe-PCIe bridge specification section 3.2.5.17 bit 6:

The bridge's secondary bus interface and any buffers between the two interfaces (primary and secondary) must be initialized back to their default state whenever this bit is set. The primary bus interface and all configuration space registers must not be affected by the setting of this bit

Let's cut it up in the relevant parts and go through an example simple hardware bridge and an example Zen bridge both initially configured for secondary bus id 8:

- 1) buffers must be cleared
- 2) interface reinitialized
- 3) config space registers may not change

For simple hardware this means that any ongoing transactions are stopped, the buffers (1) are cleared and the interface (2) is brought back to the default *state* where it is ready to train the link and start up again. Remember that the configuration registers *ARE* the configuration in this case, so after a few quick state changes the bridge returns as active with a bus id of 8 which was the same id it had before the reset - since the configuration is not allowed to change (3) and it isn't part of secondary bus interface transaction *state* anyways.

For Zen the state machine receives the reset and implements the spec to the letter: It clears any ongoing transactions from the buffers (1) and resets the underlying hardware (2). The underlying hardware also follows the spec to the letter which as per section 3.2.5.4 is to start with secondary bus id 0 after hardware reset. The config register in memory remains unchanged as 8 (3).

So technically both the simple hardware and the Zen data fabric implementation follows the specification, but the end result on the wire is different: bus id 8 vs 0. The simple hardware is incapable of being in a different configuration than its configuration registers while the Zen implementation can both reset its hardware to default while at the same time keeping configuration registers in the state machine memory unchanged.

The importance lies in the difference between the words *state* and *configuration*; and I have to admit that 3.2.5.17 is really poorly worded in this regard. I would argue that the Zen data fabric state machine ought not only reset the underlying hardware to the default *state* but also restore the actual *configuration* in it in order to properly emulate what a simple PCIe bridge controller would have done if it was not allowed to change its configuration registers. The specification states no such requirement, however, and the result is this mess we have where the hardware doesn't use the current configuration.

Ok, so what?

The solution in any case is the same: Rewrite the relevant registers. For Zen this will cause the state machine to sync back the underlying hardware to the configuration it was supposed to be in.

It doesn't really matter if it is the firmware state machine instances themselves (preferably, go go AMD), the kernel (probably, go go Gnu) or something else (like the userland Java example from OP update 6) that does the rewriting.

For simple hardware a rewrite of relevant registers to the value they already have will do exactly nothing. So it should be safe for the kernel to do this for all bridges.

Anyways, long post. Sorry that I couldn't make it shorter - am in a bit of a hurry.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)



[\[-\] AnarchoLeduc](#) • 1 point 1 month ago



Very interesting read. Thank you for taking the time to go this far into sharing your analysis.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **zir_blazer** • 6 points 5 months ago

In the months since ThreadRipper has been in the market, I think I don't recall anyone actually posting the issues with TR in VFIO Mailing List, nor if VFIO main dev Alex Williamson ever had anything to say about the matter. Did anyone bothered to contact him about TR issues? I usually don't see him going outside his bunker in the VFIO ML (He had a reddit user, but he only posted once), so you may want to drop a mail instead of expecting him to come here.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 3 points 5 months ago

Had a quick exchange with Alex and we seem very aligned on the issue. I'll be doing some tests on non-GPU devices too as a result of his feedback.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 3 points 5 months ago

That's a good idea - it will be quite interesting to know what he thinks and will be quite helpful to have him on board if a workaround (like a kernel argument for vfio-pci like "no-vga-reset") or a bugfix is to be pushed mainline.

So far there is no indication that the bug is in VFIO-PCI, though. A quick glance at the code shows that it looks fairly clean.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **lolomg** **Taichi X399, 1950X, 32GB DDR4, RX Vega 64** • 6 points 5 months ago

I have a 1950x with a RX Vega 64 and I'd love to help, but it's too advanced for me. I quite literally have zero understanding of what I just read.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 7 points 5 months ago*

Unfortunately the topic is a bit advanced - in a perfect world things would just work out of the box. Here is a small guide in case your system uses vfio-pci as a module (most distros):

- Run "uname -a" to find your current kernel version
- Use package manager to download kernel sources
- Go to the kernel dir, typically /usr/src/linux or where it was unpacked
- Copy in your current config (Google this)
- Run "make menuconfig" to verify
- Run "make -j32 && make -j32 modules" to compile a test before changing anything
- Find the file mentioned in the post above and change the two lines
- Compile again.
- Run "rmmod vfio-pci" to disable your current module
- Run "lspci -v" to check that your card is not already crashed
- Go to the folder with the new module you built (same as source of vfio-pci) and run "insmod ./vfio-pci.ko" to enable the changed module
- Try your VM

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **zukan1** • 1 point 4 months ago

I'm getting "insmod: ERROR: could not insert module drivers/vfio/pci/vfio-pci.ko: Unknown symbol in module" when trying to enable the new module. Tried recompiling multiple times just to check that I did everything right. Any suggestions?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **simcop2387** • 1 point 4 months ago

Check dmesg to see what symbol or other message that might be there. Though usually I see that when the module doesn't match the running kernel

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **lolomg** Taichi X399, 1950X, 32GB DDR4, RX Vega 64 • 1 point 4 months ago

Sent you a PM

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **YaniDubin** • 3 points 4 months ago*

Thanks so much for the work on this [/u/HyenaCheeseHeads](#). I can report that I also have this working. My setup is the Gigabyte "aorus gaming 7", 1950X, and passing through a Radeon R9 R290X. I am using the 4.14.4 kernel (which I found already has the NPT fix applied) and simply applying your reset workaround against this.

Please let me know if there is any specific testing / debug you'd like me to run on your road to coming up with a proper fix - while I have no expertise in PCI (I mostly work with embedded systems), I'm an engineer, so can get technical.

In case anyone else encounters the one issue I had (having the VM crash every time I tried to assign the GPU driver to the passed through PCI card), I realised that this was due to when I was binding the vfio-pci driver. I used to get away with letting Linux claim the GPU, then unbinding it and assigning vfio-pci when I wanted to run the VM. That is presumably resulting in the VM getting the card in a non-disabled state (same I guess as if you shutdown/restart the VM without disabling the GPU). So if you encounter this issue, that might be worth considering. Or does someone know a way to put the card in a disabled state from Linux when releasing it?

I plan to test and benchmark NVMe passthrough, and also crossfire (I have 2 R9s to pass through, and a 660 for the host). I am only going to be testing windross guests as my use case is photo processing (Linux tools not quite cutting it anymore for me and moving to a hybrid workflow) and games (where I can't run them under Linux natively or in wine).

Update:

Crossfire works fine on passthrough GPUs, performance is on par with native (as expected). In this case, Unigine Superposition benchmark 1080P Extreme preset was 4317 native, 4233 under KVM. The native test had a more favorable initial temperature (these cards hit the power/temperature ceiling so get throttled), but not sure how much that contributes to the difference. My Qemu/KVM is really not very tuned either, so may play a part.

I am more concerned with CPU/storage performance (for image processing) so will leave the graphics benchmarking as it stands. I am extremely happy with how things stand - even the GPU disable workaround is very much a setup once and forget, so very workable.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 2 points 4 months ago

Thanks for the feedback, nice to hear that it worked on that platform too.

Been working on the root problem (see updates in OP) and there is still quite a lot of work left to do. Unfortunately this next part of the debugging process is somewhat hidden inside the cpu, so progress is really slow and I'm getting really tired of the amibios screen during boot (have to reebot whenever something locks up). So right now success stories like yours is keeping me going =)

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **FlatronEZ** • 2 points 4 months ago

Would you mind running Firestrike // Cinebench native and in a VM with all ressources allocated to have some comparison regarding the overall performance? I assume many people in here would be interested in the numbers.

Also a CrystalDiskMark benchmark (native and VM) would be something making the crowd happy I suppose :P

Thanks in advance!

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **YaniDubin** • 2 points 4 months ago*

Hi [u/FlatronEZ](#), sorry it took me so long to get to this. I did have a result for native vs virtio, but really this was not workable for me, so I

knew I wasn't going to stop there. With virtio I found writes to be way too slow (an order of magnitude, both sequential and random).

I finally tested the NVMe passthrough. Unfortunately it is not as good as native, but better in most respects than virtio (passing NVMe partition through).

This is native: <https://imgur.com/YwMIDwU> This is virtio:

<https://imgur.com/oHAPGcO> This is passthrough:

<https://imgur.com/PRaOV1A>

So sequential writes are comparable to native, but sequential reads are 55-75% that of native. For random read/write it varies from 20-70%, depending on the test. So somewhat disappointing, but compared to virtio where random writes were about 1%, still a definite improvement.

Since this is simply passthrough, I'm not sure what there is to be done to improve the performance. I am hoping the sequential read/write performance might be the determining factor in working with many gigabyte photoshop files - this might have to be good enough, and hopefully it is.

Anyone else got some results from their experiences with NVMe passthrough? Any tips/tweaks?

My system is a bit of a mess right now as I only have one NVMe, and have had to boot Linux from an old ssd to do the test. Once I have my second one, and system is a bit more normal again, will try and run those other benchmarks you mentioned.

Update

Turns out the poorer sequential performance is a windross issue rather than virtualisation issue (performance sucks on a non-empty - or perhaps specifically system drive). I reran the test on a fresh empty partition (so now comparable to how the native result was achieved), and the sequential performance is on par with native. The random read/write is no better however. It could be something that tuning might help with (pinning CPUs, etc), but really not sure at this stage.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)



[[-](#)] [HyenaCheeseHeads\[S\]](#) • 1 point 4 months ago



There's now an ugly test patch available (see update 4 in OP), quite interested to see how it fares on the Aorus Gaming 7

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)



[[-](#)] [YaniDubin](#) • 1 point 4 months ago*



Great, thanks for that. I tried out the new patch, but this was a bit of a mixed bag for me unfortunately. While it undoubtedly did resolve the GPU reset issue (with startup/shutdown scripts eliminated I could start/shutdown repeatedly), I had some undesirable behaviour in windross 10.

I initially put it down to windross brokenness, however reverting to the old method resolves the main issue (it being deaf to mouse clicks, constantly spinning busy disc - but able to launch processes with start+e, start+r, etc), so now I am not sure that was a valid assumption. Various reboots prior to that did not resolve it.

Sorry for being so vague, but I'm not sure how to diagnose such issues - perhaps I should see if I can find anything in the event log now I can interact with it again. Is there any specific debug I can run in the guest to determine whether it is unhappy with PCI specifically? I didn't get any errors/etc in dmesg in the host.

Update 1

I also just had a host Xorg segfault when running the new fix and not doing anything VM related (never had that before) - after 40mins of uptime.

The error occurred in libpciaccess.so.0, so perhaps a latent issue with doing a reset of my host GPU (nvidia gtx660) using the new reset

code at bootup? I can grab a more complete error report later if that might be helpful.

Update 2

Today, none of those issues are present - so I think we may still be able to put this down to windross being temperamental. I will do further testing. The event log only really accounts for the misbehaviour of windross itself (WindowsExperienceShell.exe not responding), but does not implicate a particular driver/etc.

Will let you know how I get on with host Xorg stability.

Update 3

xorg stability was an issue that kept recurring. Even got a full system hang. However reverting to the stock kernel, I still seem to have the issue. Likely I made something unhappy when I put the nvidia card in - not sure what yet, but seems nothing to do with your patch.

So potentially all the issues I had were not related to your patch and can be disregarded entirely.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **naibaf7** • 4 points 4 months ago

Oh yes, thanks a lot. The pci.c patch works and ... everything works! Kernel 4.11+ (ROCM, AMD) and RX 480 passthrough. It even works to move the GPU between VMs and host without binding the GPU to VFIO.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 2 points 4 months ago*

Thanks for the feedback!

Also, that is a very good point, the ugly patch ended up being in the kernel pci driver instead of vfio-pci, so it is a bit more general.

Of course there is no way that it will be accepted in its current form but hopefully that can be fixed as well to make it to the kernel (missed 4.15, so probably 4.16)

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **naibaf7** • 2 points 4 months ago

Luckily I need a custom compiled kernel (ROCM enabled, with special GPU firmware files, NPT patch and now also pci quirk patch) for my work (deep learning software development on CUDA, OpenCL and HIP) anyways, so I don't mind using quirky patches for the next 12 months... Thanks a lot again!

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **FlatronEZ** • 4 points 4 months ago

Unrelated - Fix for Nvidia Error code 43

- open VM definition in virsh:
- replace the first line with:

```
<domain type='kvm' id='1' xmlns:qemu='http://libvirt.org/
```

- add the following

```
<qemu:commandline> <qemu:arg value='-cpu' /> <qemu:arg value='pc' />
just before the closing </domain>
```

save and boot VM.

Properly formatted: <https://hastebin.com/tajezapari.xml>

Cheers, I hope this helps!

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **Rhynri** • 1 point 4 months ago

Many thanks for this, you saved me some time when I get around to this.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **jezza129** • 3 points 5 months ago

I have no knowledge or hardware related to this.

Can you explain how threadripper handles I/O? I assume both zen dies have its own I/O systems. How do they overlap and not conflict?

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [\[-\] HyenaCheeseHeads\[S\]](#) • 6 points 5 months ago*

Each physical die has a number of I/O lines that can be configured for a number of nvme/sata/pcie1,2,3 host controllers.

In Threadripper there are two dies each configured with one PCIe x16, one PCIe x8 and at least one nvme. Typically one of the dies are connected to a chipset that provides additional functionality like network.

When the machine starts and enumerates all the pcie devices they all get an ID. The first part of the id is depends on which die the controller is on (numa context), the second part of the id is the bus id, the final part is the device id and subdevice id (like for a gpu with audio in it).

If you run "lspci -t" it will show you this for your system in ascii art.

Each such device typically has some way to control or interact with it. This is handled with memory regions that are allocated to it when it is found on the bus. When you write to that memory you actually write to the device and vice versa. You can see what memory regions are in use right now on your system by running "lspci -v".

The kernel is responsible for creation non-overlapping regions, but there are situations where initial setup can make this hard (for example if bios brought up a controller without assigning enough memory to it and it cannot be reset).

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [\[-\] Hull_Integrity](#) • 3 points 5 months ago

In Threadripper there are two dies each configured with one PCIe x16, one PCIe x8 and at least one nvme. Typically one of the dies are connected to a chipset that provides additional functionality like network.

This depends on how the PCI-E lanes are connected on the motherboard. On Zenith Extreme it's different, as shown here:

http://pclab.pl/zdjecia/artykuly/mbrzostek/amd_zen/threadripper/tech/zenith_pcie.jpg

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [\[-\] HyenaCheeseHeads\[S\]](#) • 6 points 5 months ago

Yes this is the beauty of Zen. It is extremely configurable.

They can also be used for socket-socket interconnects (like in Epyc) and, as of recently, can have some of the x16 ones be split into 4 x4 slots instead.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [\[-\] Hull_Integrity](#) • 3 points 5 months ago

All PCI-E lanes on TR can be split even to x1 links. The only limitation on Threadripper is that:

- UEFI has to support PCI-E bifurcation
- without separate clock generators you can set up up to 7 links, regardless of width
- with separate clock generators, like those present on Asus Hyper M.2 card, you can set up as many links as you like

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)


 [\[-\] --FuriousGeorge--](#) • 3 points 4 months ago*

UPDATED: I have a Zenith Extreme with a 1950x, and I'm also now able to boot VMs w/o a D3 hang. Awesome work.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [\[-\] Rhyndri](#) • 2 points 4 months ago


That is my setup as well. Any gotchas? I'm hoping to get this working in my Unraid at some point.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] --FuriousGeorge--](#) • 2 points 4 months ago

Not really. Change the kernel line in grub, make sure your initrd loads the vfio modules and binds vfio-pci to the gpu before the gpu driver does, and that's pretty much it.


I'm not sure about unraid, but Proxmox doesn't have accurate documentation for how to get the current kernel source (much less for how to recompile a custom version if you aren't familiar). At first I tried with a vanilla kernel that matched my version, but I couldn't get the module to load, so I went back to trying to get my distro's via git, and was able to tab complete my way to fetching the kernel without docs (it was in submodules/ubuntu-artful for the pve-kernel project, in case anyone is wondering)

That's what took the most time, and would have required some help, had I not done similar things before.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] Rhyndri](#) • 2 points 4 months ago

Nice. One advantage of unraid is it has no actual desktop [by default, at any rate] on the host (just CLI) so at least I'm not fighting the gpu driver on the host or something. I think I'll be able to sort out the kernel sources on unraid with help from that forum, but thanks for letting me know, the Zenith is a little bit of an odd duck.

Also, I hate the bios fan control. It's not very adaptable. You can't, for example, control the CPU fan with the Water temp probe, or set a PWM fan below what the board has calibrated is the minimum. Other than that I like it.


[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] --FuriousGeorge--](#) • 2 points 4 months ago*

Proxmox doesn't come with a gui either, nor does the nvidia-driver package from the debian repo it uses work (can't find headers on proxmox). I just like the option of having a gui since it's for a home/office lab, i.e. not a real server. I may even go back to using it as my primary desktop. I like how linux implements virtual desktops better than how Windows finally does.

As to the bios, I used to overclock, but only with air, and I didn't do much with fans other than run them full blast. I don't really have frame of reference for other boards.

I am using the fan control now, since this is in my LR, and I notice it's nice but limited. E.g. it can auto calibrate voltages, but you can't override the min. duty cycle for any fan, other than turn it off. So it can be off, but not at 20% if your min. duty cycle is 35%.

I can say that I like that the mb has vertically oriented m.2 And one under s. bridge that all use cpu lanes. Tests have shown that placing them by the pcie bays leads to near instant throttling when the adjacent gpu is under load. This is especially true on boards that have one between the cpu and 1st gpu.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] Rhyndri](#) • 2 points 4 months ago

Ah, I see. I'm not familiar with Proxmox at all, my bad!

Yeah, that fan bit drives me nuts. And I don't know if it's just my bios revision, but I don't get the option to shut them off unless I'm using DC and not PWM. Which means you actually can't use the off feature on certain PWM fans (like the EK Vardar). I like my systems to be silent or near it when not under high stress, so I've had to work around these bios limitations; on my previous machine I just used

Speedfan under Windows for control, which doesn't have any of these limitations, I even had the GPU blower on my 670 hooked up to it and off when the card was in 2D mode.

In regards to the M.2 support, it is excellent. I did wind up moving mine to a Hyper M.2 card for the cooling, although I'm sure my Dimm.2 will eventually be in use again, probably for slower larger capacity storage.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

⬆️ [\[-\]](#) **Marty_Br** • 2 points 5 months ago*

⬇️ I'll be happy to toy with this. I mentioned this before, but I managed to get my system set up to where I am able to run a VM with passthrough while being able to shutdown and restart or reboot at will. I described it in another post.

In my particular case, it works when I load vfio-pci with 'disable_idle_d3=1' and run the system startup script to enable the GPU in the VM. Note that the shutdown script on my machine causes a catastrophic host crash (with file system corruption). I do not currently run that script, obviously.

The issue I now encounter is that if I run the VM and then shut it down, after 20-30 minutes or so, the GPU will suddenly run its fans at absolutely maximum speed (like a tornado). I think it overheats. So while I am able to either reboot or shutdown the VM and then restart at some later point, there is still a serious issue that I have not been able to resolve. Edit: when that Vega goes hog, it also becomes unresponsive even with host reboot. It requires a complete power off cycle.

For completeness: I have a 1950X running on a Gigabyte Aorus 7 with 32 GB @ 2666. I am running Ubuntu 17.10 with a v4.15-rc1 kernel with ACS patch (for USB passthrough for my Vive). GPU 1 is an Evga 1070 and GPU 2 (passthrough) is a Vega RX 56.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

⬆️ [\[-\]](#) **HyenaCheeseHeads[S]** • 4 points 5 months ago

⬇️ Vega is a bit of a special case. The OP was already quite long so I skipped it but hopefully the workaround should work for you too. Here is why:

It seems there is some issue with the Vega cards so that they either do not report hot-reset capability or ignore the command when it is sent. In other words they do exactly what the first part of the workaround above does: they ignore hot reset. So you pretty much get the first line for free, so to speak.

BUT

The second part of the workaround has to do with bus-resets and are not card-related at all. When I had this line active I experienced some of the same issues as you with many of the cards - closing down VMs would eventually lead to catastrophic failure of the host OS, typically with varying amounts of PCIe AER/ACS warnings being spewed in dmesg.

With the erroneous bus reset it will look like the CPU disconnected from the PCIe bus. Since the driver is partly responsible for fan control (keeping the fan low) you will hear the card timeout reset where it goes back to the standard BIOS fan setting of 100% after a while.

Please do try out disabling both lines and see if this fixes your issues - even with Vega

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

⬆️ [\[-\]](#) **Marty_Br** • 7 points 4 months ago

⬇️ Well, what can I say. I commented out those lines, and compiled and installed new kernel and modules. Result: this works! I put back the shutdown script in my VM to properly shut down the GPU -- which prevents it from going crazy after a while when the VM has been shut down -- and this now results in neither a crash nor a 7F header issue.

As such, it looks like I now have a system that works completely.
Now to go locate those Ryzen SMT patches for qemu ...

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **jipe** • 1 point 4 months ago

▼ This is pretty great to read as to me it means that there's hope for the TR platform yet when it comes to PCIe passthrough. I'm just wondering if this would apply to using KVM on any ThreadRipper main board? Because if it is applicable to any TR board, my last reason not to buy one just imploded.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **Marty_Br** • 1 point 4 months ago

▼ I cannot really comment on any boards other than the one I have, but I can say that the particular board I have is one that a number of other posters have had particular difficulty with: this board is not known to be the easiest one to work with. With the current kernel patches, setting up KVM is almost trivially easy.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **HyenaCheeseHeads[S]** • 1 point 4 months ago

▼ If you happen to be up for another round of testing there is now an ugly patch (see update 4 in OP). I'm really curious to hear if this works with Vega. If we are really lucky it may also fix the shutdown issues and allow you to run without scripts in the VM. If you do try out the ugly patch, please first revert the current changes and confirm that the bug is back before applying the patch.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **Marty_Br** • 1 point 4 months ago

▼ Well, Mr. hyenacheeseheads, I will be more than happy to give that a go. The previous solution actually solved my issues completely, however. I currently have a setup that just works. I will, however, test this one and report back to you.

M

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **Marty_Br** • 1 point 2 months ago

▼ Alright. I did try the "ugly patch," which did work. Very nicely, in fact. Unfortunately, my windows virtual machine installed a set of updates this weekend, and now it no longer appears to be working. Very frustrating.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **Marty_Br** • 1 point 5 months ago

▼ Thank you. I will toy with this and report back. It would be nice to get this to work.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **Lithium2142** • 3 points 5 months ago

▼ The fans going full blast sounds like a GPU hang. On windows the GPU will reset if unresponsive for a certain time. This still isn't implemented on Linux yet.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

^ [-] **bsodmike** • 1 point 4 months ago

▼ Marty - how did you get on with installing NVidia drivers for v4.15-rc1? In Fedora 26 I've not had much luck...

<https://forum.level1techs.com/t/errors-when-installing-nvidia-drivers-for-kernel-compiled-from-source-4-15-rc1-for-fedora-26/121994/3?u=bsodmike>

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Marty_Br** • 1 point 4 months ago

bsodmike,

I'm on Wayland, so the Nvidia drivers aren't an option at all for me right now. Frankly, it's a complete mess. It'll get sorted in time.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **ChIPixo** • 2 points 5 months ago

I have observed same behaviour on the Ryzen.

Bus reset for the SATA and Audio resulted in the same PCI malformed header. If i prevented bus reset by not binding the SATA to vfio-pci all was working. Strange thing is that if i passed Vega 64 i had this malformed header after reboot or shutdown. The R9 290 i had been using before it newer had this problem.

Observed the Audio and SATA on 2 different Ryzen 1700.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **irhaenin** • 2 points 4 months ago

Wow, you really did some impressive digging. I've been holding off on buying TR because of this issue, but it begins to sound like you pretty much managed to fix this issue in software.

I'm curious about a few things though. You mentioned seeing the infamous 43 error when using the no-reset workaround. However, if you restore the PCIe bridge configuration space after issuing a reset, do you still get the error, or are you able to pass through NVIDIA cards successfully as well?

Furthermore, could this issue be related to, what I assume is, the fact that TR with its 2 CPUs in a single package has 2 PCIe bridges/buses. As in, could the configuration of bridge A somehow be polluting the configuration of bridge B? I'm merely speculating here.

Thanks for taking so much time to fix this.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 2 points 4 months ago*

On the superglue:

Really good question! Initially that was the working theory in the tests too. However, there are a number of items going against that theory:

- The issue happens both on slots connected to side A and B
- The bridges themselves seem to respond just fine and it is possible to differentiate between them
- Although fairly new in the consumer market, this kind of setup (multiple PCIe roots) has been used before in server products
- Both the GMII/Infinity Fabric that connects the two dies and root-to-bridge connections seem reasonably stable. You would expect more stuff to go wrong if there was some kind of clash between the cores that could cause config from one to end up on the other.

An example Threadripper pcie bus layout:

```
~ $ lspci -t
+-[0000:40] +-00.0
|          +-00.2
|          +-01.0
|          +-01.1-[41] ---00.0
|          +-01.3-[42] --+00.0
|          |          \-00.1
|          +-02.0
|          +-03.0
|          +-03.1-[43] --+00.0
|          |          \-00.1
|          +-04.0
|          +-07.0
|          +-07.1-[44] --+00.0
|          |          +-00.2
|          |          \-00.3
```



```

|          +-08.0
|          \-08.1-[45]---+-00.0
|          \-00.2
\-[0000:00]---+-00.0
          +-00.2
          +-01.0
          +-01.1-[01-07]---+-00.0
          |          +-00.1
          |          \-00.2-[02-07]---+-00.0-[
          |          +-04.0-[
          |          +-05.0-[
          |          +-06.0-[
          |          \-07.0-[
          +-01.3-[08]---+-00.0
          |          \-00.1
          +-02.0
          +-03.0
          +-04.0
          +-07.0
          ...

```

At the top level you see the two sides. Each has a number of devices and bridges beneath it. The chipset is also easily identified since it is itself a bridge with more devices (like ethernet) connected to it. In this example both bridge 0000:00:01.3 and 0000:40:01.3 have gpus attached. The gpus provide 2 functions (00.0 and 00.1) because they also have audio circuitry in them.

You COULD be right, though. If we could read the config registers directly from the hardware it would be easy to see if it was the case.

Unfortunately I'm a bit of a chicken when it comes to soldering JTAG interfaces to expensive hardware, and really wouldn't have much of a clue what to do anyways. Better leave that to AMD engineers when it comes to it.

NVidia:

This issue does not seem to be specific to AMD/Nvidia or even GPUs at all. I just don't like working on Nvidia cards when they actively try to work against virtualization in their driver by throwing an error or a bluescreen when they detect it. Working around that check is possible but outside the scope of this thread. There should be no difference between the workaround and the real fix in that regard. There was another redditor further up the thread that confirmed a working 1080ti.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **duidalus** • 2 points 4 months ago

I can confirm this works as well on TR1950X and Linux 4.14.5. One thing worth noting is that SME caused some weird errors on my system when trying to passthrough anything.

To ensure everything works as expected, use mem_encrypt=off as boot parameter.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **LupaioliS** • 2 points 4 months ago*

I can't get the "ugly patch" to work on TR 1950X on Asus x399-a in Unraid (R9 290).

Any suggestions?

Update:

I tested with the workaround commenting lines and there is no more D3 state report on the log but still black screen and "internal error: Unknown PCI header type '127'" when starting the VM.

Update 2:

I got my VM to work with the R9 290, i was putting the wrong flag on boot.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [-] **noc0lour** • 2 points 4 months ago

Can confirm on 4.14.4 with 1900X, Gigabyte X399 passing through a RX580 works now. Didn't work without the patch. The reinit issue is gone as well. I'll be happy to provide you with more info if you need it. Thanks for investing your time in this!

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [-] **noc0lour** • 1 point 4 months ago

Though I didn't see the UEFI Splash screen again from Tianocore after the first boot.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [-] **rcgoodfellow** • 2 points 3 months ago

Dirty patch works for me. Thanks!

- Fedora 27 - patched kernel from origin/f27 branch (4.14.13-300)
- ASRock X399 Pro Gaming
- Radeon WX 5100 on host
- Radeon RX 550 Passthrough
- Windows 10 Guest

Tested with TF2 running simultaneously on host and guest.

fwiw: OVMF stable in Fedora package repos did not work, had to use latest build from git. Also passed through USB for keyboard, mouse and Bluetooth dongle for XBox One wireless controller with success.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [-] **simcop2387** • 1 point 3 months ago

One thing I found out about this MB. Every PCIe slot except the 1x one is on its own iommu group. The 1x ends shared with about 8 other devices on the MB in group 12. No idea how nvme looks for pass through

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

 [-] **w23** • 2 points 3 months ago

Thank you so much for this! I've been waiting resolution for this problem for months, and after seeing this I finally went and bought a TR box.

My setup:

- Threadripper 1950x
- ASUS Prime X399-A, stock BIOS 0318
- Host GPU: AMD Radeon R9 FURY X
- Guest GPU: NVIDIA GTX TITAN 6Gb
- Kernel: 4.14.12-gentoo
- Qemu: 2.10.1
- OVMF: 2017_pre20170505 (edk2 git commit f30c40618b1f3537705b450a91ce00b9e587badb)
- Guest OS: Windows 10

IOMMU groups + reset flag: <https://pastebin.com/B5FkadL7>

Status:

- QEMU+OVMF+Q35 works flawlessly with this patch. Haven't even tried without.
- Even though no reset method detected for guest GPU, VM has no trouble restarting from any condition (e.g. even being Ctrl+C'd) without any reset workarounds.
- QEMU+SeaBIOS+Q35 doesn't work (tried the very same VM and startup script that worked for previous box): GPU doesn't initialize, monitor doesn't detect any signal. After VM is converted to UEFI+GPT (booting from WinPE iso and using native mbr2gpt.exe utility), it works just fine with OVMF.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

 [-] **droidmahn** • 1 point 4 months ago

Can confirm this works as described on a 1080ti guest. Ill do more in depth testing in the AM.

Specs for me: 1950x Asrock x399 taichi 1080ti host gpu 1080ti guest gpu Was able to get as far as installing windows which is what its doing now. Ill update test tomorrow

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **jipe** • 1 point 4 months ago

Nice! could you please let me know if e.g. gaming works? I'm looking to buy a similar setup, same CPU, mobo and guest GPU.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 3 points 4 months ago*

I've tested the Unigine Valley benchmark, Firestrike and Timespy benchmarks as well as S.M.I.T.E. - it looks like they work just fine when both the NPT patch and the busreset patch is applied and the card has a screen or an EDID-faker in one of its output ports.

Furthermore I've gotten a bit further in the identification of the root cause of this issue - in fact much closer to an actual fix. Today it was possible to bring a card back from the all-FF disconnected state that they crash into, this time without having to reset the host. Haven't had time to update the OP with the new info yet but will do that before next weekend.

Working with Alex (vfio maintainer) to figure out in what direction to go.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **droidmahn** • 1 point 4 months ago

Yeah may take me a bit to get to that point. I'm waiting on gnif to release his frame relay so I dont have to buy another display lol

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **zukan1** • 1 point 4 months ago*

I got the patches working but unable to get past the error 43 code. How did you get around it? None of the quick fixes in the xml file seems to work for me.

I have been trying with both a Nvidia 780Ti and a 970 on a 1920x Asus Prime x399-A. In a Win10 guest I see error 43 in the device manager, whilst in a MacOS guest it won't boot and dmesg is filled with BAR 3 errors.

UPDATE: I managed to get it working with the latest patch provided by OP. Everything OK now on both GPU's. Amazing work HyenaCheeseHeads!

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **PetebLazar** • 1 point 4 months ago

Ugly patch works for me too.

ASRock X399 Taichi & 1950X (HostOS: Xubuntu 17.10x64 with vanilla kernel 4.14.10 + "ugly patch") Host GPU: 1080Ti, Guest GPU: GTX 960 (GuestOS: Windows 8.1 x64)

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **zAxny32** **ASUS X399-A, 1900X, 32GB DDR4 3200, GTX 1080 Ti** • 1 point 4 months ago*

I can confirm that your patch on update 4 works with my threadripper 1900X on linux 4.13

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **--FuriousGeorge--** • 1 point 4 months ago

Dirty patch worked for me.

Zenith Extreme / 1950x / Proxmox w/ Kernel 4.13.13, 1080ti, 1070, 970.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **Tree_Mage** **TR 1920X | RX 580** • 1 point 4 months ago*

System specs:

- Asus ROG Strix
- TR 1920x
- Passthrough GPU: RX580
- Windows 10
- Kernel 4.15rc4 (linux-mainline on Arch)

Before: I could sometimes get vfio to work.... only after I waited an hour or two for the VM to actually boot. Very unreliable.

Post-pci_reset_secondary_bus 2017-12-11 ugly patch:

I'll be out the rest of the week, so I won't really have a chance to stress test it. But so far, it looks promising. It now boots in a minute or two (slow drive).

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **Timelord0** • 1 point 4 months ago

I have applied the patch in the latest version of Proxmox, seems to fix the issue well enough. Hardware 1950X, Zenith Extreme, GTX1070 w/EFI capable firmware.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **rv6502** • 1 point 4 months ago

I've tried the Update 4 "ugly patch" (no other changes) and I can now launch QEMU with GPU passthrough but the guest Linux (xubuntu 4.13.0-16-generic) is unable to use the GPU at all.

With a R7 260X the guest FOSS Radeon driver crashes the guest kernel on boot trying to initialise the card. I'll boot if I use nomodeset and the card shows up in lspci but won't activate.

With a nvidia 8600, neither the nouveau driver nor the closed source work. I get no crash but the card does not respond and after trying once the card will not show up anymore after host reset until I power cycle the host machine.

1950X on MSI X399 gaming pro carbon ac.

Host OS is xubuntu 4.13.0-21-generic.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 1 point 4 months ago

Interesting

Is this the same with the original workaround?

Do you see any output at all (including an active signal with just black) from the passthru card on the host *before* starting the VM?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **rv6502** • 1 point 4 months ago*

I haven't tried the previous workarounds where the reset gets commented out.

The card is the secondary card and assigned to pci.stub so it doesn't get used until the VM starts. The MSI X399 BIOS doesn't use it.

And the Linux kernel on the host needs nomodeset otherwise the system will crash on boot so it doesn't get used here either.

If I don't pass nomodeset both card shows the xubuntu loading screen animating (so I know the 2nd card works) during the startup then the secondary card goes black and the system crashes. From what I understand it's a bug with Nouveau on 10-series. I might not need nomodeset now that I installed the nvidia proprietary drivers.

But nouveau-driver issue aside I know the R7 260X card can activate and output a signal outside the VM.

Does the other motherboard BIOSes (even if temporarily) activate the secondary card? That could be what makes it work.

Edit: the primary (used by host) card is a 1050

Edit: Screenshot of the crash in the guest:

<https://imgur.com/VJQumvh> (loading xubuntu-17.10-desktop-

amd64.iso)

Edit: I just tried without nomodeset, the host boots fine but still no luck with the Linux guest on the passthrough. Currently downloading Win10 to try if it has better luck with the GPU initialisation.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **rv6502** • 1 point 4 months ago

Update: I've installed Win10 with the Radeon drivers and the GPU activated fine in Windows.

So evidently the "ugly patch" is enough for a Win10 guest but not quite enough for a Linux guest.

Which is good enough for me for now but I'll find time for more testing if you have further fixes to get Linux going as guest. Feel free to ping me.

Thanks.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 1 point 4 months ago

It would definitely be interesting to know if using vfio-pci and the no-bus-reset workaround from the OP works in your case.

While the patch from update 4 tries to fix the error after it has happened the original workaround tries to avoid triggering the issue in the first place.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **rv6502** • 1 point 3 months ago

I can relaunch the VM if I disable the GPU before shutting down Win10.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **PetebLazar** • 1 point 3 months ago

Ugly patch work for me (ASRock X399/BIOS2.00. 1950X)

HostOS(GTX1080Ti): Xubuntu 17.10x64 GuestOS(GTX960):
Windows 8.1x64, Windows 10x64, Xubuntu 17.10x64

http://www.monitos.cz/tmp/GuestOS_Xubuntu_GFX960.png

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **sudoprime** • 1 point 3 months ago

Thanks for this, you saved AMD a sale! I am now getting video out of my pcie passthrough. Though I'm hitting error 43 in the guest now with my first gen titan x :(

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **Teknoman117**

Threadripper 1950X | Gigabyte X399 Aorus | GTX 1080 Ti | PG348Q • 1 point 3 months ago

did you specify kvm=off in your qemu line (or hypervisor=off or whatever the libvirt equivalent was) ?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **sudoprime** • 1 point 3 months ago

yeah, I did a whole bunch of crap, this patch, the npt one, vfio devices in my grub. KVM off in my libvirt conf, etc. It was a pain in the ass.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Teknoman117**

Threadripper 1950X | Gigabyte X399 Aorus | GTX 1080 Ti | PG348Q • 1 point 3 months ago

personally going to try and give this a whirl when I get home -
<https://github.com/sk1080/nvidia-kvm-patcher>.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Teknoman117** Threadripper 1950X | Gigabyte X399 Aorus | GTX 1080 Ti | PG348Q • 1 point 3 months ago

If you need anymore testers, I'll be getting my gentoo install stood up in a week or so.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **simcop2387** • 1 point 3 months ago

Wanted to chime in and say that I got my system built this weekend and tested. The ugly patch seems to work great for me on 4.14.12. I haven't seen any side effects from testing and it seems to work great with my two testing cards (older radeon ones).

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **simcop2387** • 1 point 3 months ago

Also confirming that it works with a GTX 1070.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Yukicanis** • 1 point 3 months ago

I am planning on getting an AMD Threadripper 1900X soon. I wanted to use the MSI X399 SLI Plus, since it is the cheapest motherboard currently available and has all the features I need. Has anybody already tested it?

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **rv6502** • 2 points 3 months ago

I got a 1950X on MSI X399 gaming pro carbon ac which by the looks of it is the same motherboard but with added aesthetic bling-bling for an extra \$10 (Which I put in an old steel server case with no window so you can't see anything, they were sold out of the cheaper one when I bought it :P)

I've only tested the 4th update patch so far and I can run a Win10 guest once with a Radeon R7 260X card after that I need to power cycle the host machine, Linux doesn't work as guest for me with the GPU passthrough.

I haven't had the time to test the other patches yet so it's still promising.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **rv6502** • 2 points 3 months ago*

I tested it again. I can get the Win10 VM to shutdown and restart with the GPU but only if I disable it before shutting down the VM.

I misunderstood and thought the 4th update's "ugly patch" didn't require disabling the card before shutting down the VM.

So it looks like the MSI X399 works just as well as other ThreadRipper motherboards as far as GPU Passthrough goes.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Yukicanis** • 1 point 3 months ago

Thank you! :)

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **zukan1** • 1 point 3 months ago*

Just want to confirm the Java patch works fine here on unmodified 4.14.12 kernel, thanks OP!

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **podkill** 1950X, 32GB, Asus GTX 1080ti oc • 1 point 3 months ago

Thank you for all the effort you put into researching this.

The "java hack" works for me until it gets fixed another way.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 1 point 3 months ago

Thanks for the feedback! Just for reference, what is your system specs?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **podkill** **1950X, 32GB, Asus GTX 1080ti oc** • 1 point 3 months ago*

OS: Arch Linux
Kernel: x86_64 Linux 4.14.13-1
CPU: Threadripper 1950x
Motherboard: MSI X399 Sli Plus, BIOS 7B09vA2
RAM: Crucial DIMM 32 GB DDR4-2666
Host GPU: Radeon RX 560
Guest GPU: Asus 11GB D5X GTX 1080 Ti STRIX OC GAMING
Qemu: <https://github.com/spheerik/qemu.git> (for glitchfree audio)
Guest OS: Windows 10

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **Kevadu** • 1 point 3 months ago

I tried the ugly patch on my TR 1950x/ASRock x399 Taichi/Vega 64 (guest) system and it didn't seem to change much. Mind you it didn't hurt anything either but it's functioning just as it did before: passthrough works the first time but if I shut down that guest OS nothing else can use the card.

That said I know that Vega has its own reset issues that are likely unrelated. People seem to experience the same problem even if it's connected to an Intel system. So I just wanted to ask if anyone here has gotten this to work with a Vega card.

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **d9c3l** • 1 point 2 months ago

Since the patch from [/u/gnif2](#) doesn't work (the gpu is still not being reset), I decided to try to give your "java hack" a try but it doesn't work as well (it sees the pci devices bound to vfio-pci but when it reads the 'config' the first 4 bytes are only 0xFF and not being checked). Do you have any opinions about it?

[permalink](#) [embed](#) [save](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 1 point 2 months ago*

What does the entire output look like?

What is your kernel version and hardware?

"The Java hack" program continuously scans the first 4 bytes from the config of the card to detect when it needs to rewrite the entire config of the bridge (2 different configs in play here). If those 4 bytes from the card are 0xFF (which is an impossible value for them to be, since they are supposed to be hardware and vendor IDs) then it triggers the rewrite of the bridge responsible for the bus that the card is connected to immediately - since obviously something must have gone wrong at the hardware level for this to happen.

Did I understand your question correctly?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **d9c3l** • 1 point 2 months ago

The output of java program? If so its <https://ghostbin.com/paste/noc3e>. I did restart after using qemu since the gpu didn't reset, and it looks like the config did return to normal but the java hack is still giving that output (even after changing the bytes to match what's in the config which is 0x02 0x10 0x63 0x68).

Kernel: 4.15.2-2-ARCH (with [/u/gnif2](#) patch currently).

CPU: AMD Threadripper 1950x

MB: asus x399 zenith extreme.

GPU (Guest): amd vega frontier edition (air cooled).

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)

[-] **HyenaCheeseHeads[S]** • 1 point 2 months ago*

Odd, according to the program you are using a non-Zen based bridge. I put in that check specifically to make sure that the program would do nothing if used on a different chip without the error described in the OP.

The program output states that it detected your passthru device but skipped it and is monitoring 0 devices.

Maybe you have a different version of the bridge? That sounds really interesting!

- 1) what is the output of "lspci -tv" and "lspci -vvvn" right now?
- 2) move the card to other slots and try again with an unmodified version of "the Java hack" to see if it recognizes your bridge

One more thing: Vega is a bit of a special beast. It may have other problems than what is being discussed in this thread. You may actually be able to solve it using "the Java hack" with a Zen bridge but there are no guarantees since I never managed to get my hands on one.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)



[\[-\]](#) **d9c3l** • 1 point 2 months ago*

Maybe asus is doing something weird with this motherboard. Hopefully thats not the case.

lspci -tv output: <https://ghostbin.com/paste/bojmf>

lspci -vvvn output: <https://ghostbin.com/paste/3nenw>

I will move the card to another slot when I get home later and give you an update. Might swap put it into the first slot and see if that helps. I have heard there were some issues with vega but also heard most was resolved in 4.14 with more fixes coming in 4.16.

Also, I forgot to mention that in `/sys/devices/pci0000:00/0000:00:03.1/0000:09:00.0/0000:0a:00.0/config` that the first four bytes are 0x22 0x10 0x71 0x14 since the "java hack" is checking the bridge config. I believe I was checking the device config so I will get back with you on that too.

UPDATE: I do believe its a different version of the bridge. The bytes I mentioned before in the other comment were from the device, not the bridge.

After doing a quick test with that small change, the "java hack" do see the bridge. Booted up qemu and then shut it down and I do see the program is attempting to reset the bridge but it is failing to recover. the value of `/sys/devices/pci0000:00/0000:00:03.1/0000:09:00.0/0000:0a:00.0/0000:0b:00.0/config` is all 0xFF except for the last byte which is 0x0a.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#)



[\[-\]](#) **HyenaCheeseHeads[S]** • 1 point 2 months ago*


Ok so there's definitely another bridge inbetween the Zen bridge and the actual GPU hardware but it looks like it may be physically located on the graphics card. That is a quite non-standard, but a perfectly valid configuration. Moving the card will probably not help with this as the extra bridge moves around with it.

Hm.. "the Java hack" will incorrectly be targetting this inbetween bridge because it just assumes that the error is at the closest bridge to the GPU.

If you take the unmodified program and on line 56 where it says `bridgePath = devicePath.getParent();` and add some extra `getParent()` like this:

```
Path bridgePath = devicePath.getParent
```

... somewhere between 2-4 probably. Does it then detect the Zen bridge and reset it? I wonder wich bridge is failing... could you post another `lspci -vvvn` from while the card is not working?

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] d9c3l](#) • 1 point 2 months ago*



It does detect afterwards but it would still not recover after adding that.

Here is the output:

<https://ghostbin.com/paste/54fxp>

In that output, the gpu does show up as "!!!
Unknown header type 7f"

Just to give an additional statement, it took me a while because when I was attempting to see if it would act up again, the gpu was being passed through successfully. This happens at random, but even after closing the vm, there was no change in anything so I am unsure if the device was being reset or not at that time, but now it wont let me boot the vm unless I restart the machine. Quite confusing honestly. Makes me wonder if there is more to it

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] HyenaCheeseHeads\[S\]](#) • 1 point 2 months ago*

From that output it does indeed look like the Zen bridge is doing alright and it is the other bridge - the one on the card - that is acting up. Very interesting, as this other bridge is also an AMD bridge (based on vendor id).



If you shut down your computer, unplug power for 30s, start it back up and boot directly into Linux without starting the VM, could you then take a copy of the configs:

```
cp /sys/devices/pci0000:00/000
cp /sys/devices/pci0000:00/000
cp /sys/devices/pci0000:00/000
cp /sys/devices/pci0000:00/000
```

Then start and stop the VM and make sure the "Unknown header" is in lspci and take another copy into another set of 4 files.
Then do

```
cp ./ok.2.conf /sys/devices/pc
```


to restore the device's parent bridge original config from boot and check with lspci if the GPU is back. Regardless of result please post all 8 files somewhere on the web.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] d9c3l](#) • 1 point 2 months ago

I did copy it back over but that did not restore the device.

The files are here

https://nofile.io/f/jut4dxm0Wr7/pci_dev.tar.xz with the sha256 being
77cb81e1e2942d0ffb2e600ceceb239
d7c20e6260c904e3838ce044b24ea0
31b

[permalink](#) [embed](#) [save](#) [parent](#) [report](#) [reply](#) [\[-\] HyenaCheeseHeads\[S\]](#) • 1 point 1 month ago

Sorry for missing your reply.

In your case it is clear that the bridge on the card is messing up (the card goes 0xFF but the bridge is still there). It looks very similar to the Threadripper issue but it doesn't respond to the same fix.

Something else is going wrong.

[permalink](#) [embed](#) [save](#) [parent](#) [report](#)
[reply](#)

[continue this thread](#) →

[+] **LightTracer** *comment score below threshold* (13 children)



about	help	apps & tools	<3
blog about advertising careers	site rules Reddit help center wiki reddiquette mod guidelines contact us	Reddit for iPhone Reddit for Android mobile website buttons	reddit gold redditgifts