Hardware Performance

Benching, Render Efficiencies, Multi-Screen Playback

Rendering

Rendering

Bench Tests

Since 2017 or so, I've been running **GeekBench** statistics on most machines I've custom built or buy pre-built. Since GeekBench measuring standards updates fairly often, I always have a level set where I've got a device benched in the latest version and the previous version. Everything is proportional, so I can make some assumptions that way. I also will run renders on whatever machines I have available to me every few months to see which performs best. It's a good level set.

You'll notice that I don't run a Maxon based Cinebench because I don't do a lot of 3D work and Apple devices don't have NVIDIA GPUs so they cannot be evaluated for CUDA. In general, dollar for dollar, Windows boxes out perform Apple here to a massive degree. I will run Cinebench to evaluate if a media server is having issues with a GPU, or if I'm just curious.

Latest and Greatest Render Tests (ca. 2024)

128,000,000 pixel source composition grid file, split into 5 renders of different sizes. This resolution in exponential K is closest to 16k, but if you measure by number of 1920x1080 that fit into it, it's the equivalent of about 61k.

Competition

- M3 Max Mac OS Sonoma with 16-Core CPU and 40-core GPU / 128GB Ram / 2TB / w/ 500GB Cache
  - GeekBench 6 Single-core: 3193 / Multi-core: 21393 / Compute: 154180
- M1 Max Mac OS Sonoma with 10-core CPU and 32-core GPU / 64GB Ram / 4TB w/ 400GB Cache
  - GeekBench 6 Single-core: 2438 / Multi-core: 12769 / Compute: 120302
- Boxx Win 10 Pro Intel i9-12900K 16-Core (24 logical) 2x RTXA4000 16GB / 128GB Ram / 2TB OS / 4TB w/ 450GB Cache on a secondary internal M2
  - GeekBench 6 Single-core: 2684 / Multi-core: 15422 / Compute: 121267

Results

The M1 out-performed the Boxx machine by a factor of 2:1 when initially rendering these files out of After Effects directly to Apple Pro Res 422. Frame rendering into an image sequence (PNG, TIFF, etc) performed at the same proportion. If I wanted to re-assemble those breakouts into combination files, the Boxx machine outperforms the M1 by a factor of 2:1. The M3 functioned proportionally to that.

As expected, rendering as a bench test is variable based on what you're rendering. Even if it's as simple as combining files or splitting them up. That the M1 outperforms the Boxx in any rendering

bench is very good. For most of the last two decades, you'd never see an Apple device do this.

Running splits or "tiles" of rendered content via FFMPEG, the M3 outperforms the M1 and the Boxx by a factor of 2.5:1 in some cases. Pretty wild.

Render Efficiencies

Working in Creative Technology means you'll probably be rendering high-resolution, oddly sized media in production settings where every second matters. Over the years, I've developed some pretty good tricks to speed things up, both in processing, and in workflow!

Whatever program you work the fastest in, that produces the media you need to produce, is the program you should work in when you're under pressure. When you're not under pressure, you should figure out what programs do what things better for processing time.

Premiere can render edits and color much faster than the identical edits and color in After Effects – so for editing, use Premiere. But, After Effects has much more opportunities for automation and custom workflows, so for complex image processing, use After Effects and then bring it into Premiere.

Remember that efficiencies are only important at scale – what I mean by that is that if you're just concepting something out and doing it dirty, just use whatever you can to get the concept at of your brain and into napkin-mode, but as soon as you're making that thing in a production model, efficiencies begin to matter. You'll see throughout this handbook that I've used Vectorworks for diagrams; this isn't the best program for pixel layouts, but it's the one I'm fastest in!

Rendering

Tiles & Splits: A Projector Blend Rendering Case Study

This case study assumes some understanding of projection engineering and that the blend is happening on the Media Server side. It is possible to bake blends, but it's not something I recommend unless you have to.

Splitting content up after a content render is more efficient then rendering splits at the source.

Say that I'm going to be projecting a 3x1 4k projector array at a 25% blend. This means that my overall canvas is 9600 px wide x 2160 px high. You could call this my "content canvas." A template for creative folks would just be a blank canvas at that resolution. Typically, at higher resolutions, you may need to separate out your media into "splits" or "tiles" so that you are playing back efficiently on your show machine.

The obvious thing to do would be to configure your template so that you render out the slices from there. This is the easiest, less thinking required, method to rendering splits – but it'll take a ton of extra time as the computer needs to process the whole canvas three times, instead of once! In some cases, say in Premiere, that might make sense – but in After Effects, this could really add a lot of render time, particular to high resolution renders.

Instead of processing your composition three times, you render it once and use a different tool to process it into splits. It requires less thinking to process your composition three times, but it requires less time to process it into splits as a post-creative-render process. Bottom line: It is faster to render the whole surface than pre-render splits in After Effects.

For doing this efficiently, I recommend FFMPEG – it absolutely rips through a splits process. That said – you can achieve similar ends using Premiere (but will require more QC, as there are greater human error possibilities), or Shutter Encoder (a great GUI for FFMPEG).

Tiles-And-Splits-01.png

Below is an FFMPEG script that will take your source media at 9600 wide x 2160 high and split it into the 3 discrete projector outputs, including a 25% 960px overlap. It will not change the duration or change the quality provided that the source is Apple Pro Res 422. **If there is any junk in the original encode, FFMPEG will inherit that junk "garbage in, garbage out" (i.e., non-square pixel aspect ratio, naughty frame rate, etc)**.

```
ffmpeg -i /Volumes/filepath/source9600x2160.mov -filter:v "crop=3840:2160:0:0"
-c:v prores_ks -profile:v 3 -pix_fmt yuv422p10le /Volumes/someotherfilepath/splitA.mov &&
ffmpeg -i /Volumes/filepath/source9600x2160.mov -filter:v "crop=3840:2160:2880:0"
```

```
-c:v prores_ks -profile:v 3 -pix_fmt yuv422p10le /Volumes/someotherfilepath/splitB.mov &&
ffmpeg -i /Volumes/filepath/source9600x2160.mov -filter:v "crop=3840:2160:5760:0"
-c:v prores_ks -profile:v 3 -pix_fmt yuv422p10le /Volumes/someotherfilepath/splitC.mov
```

To explain the script, the crop variable has four values – the first is the cropped width, the second is the cropped height, the third is the horizontal position of the top-left pixel of that crop, and the fourth is the vertical position of the top-left pixel of that crop. **So for split B** (`/Volumes/someotherfilepath/splitB.mov`), the output is 3840x2160, and we're cropping from the original canvas where our top left pixel start is at pixel 2880 across and pixel 0 at the top. The diagrams over the next few pages illustrate how this works.

You can steal this script and use a spreadsheet with formulas so that you can change dimensions, the crops, the number of tiles, the source file location and name, and the exported file location and name.

If you get good with this, you're worth your day-rate and more. Most production companies will use After Effects to run their splits because it can be automated – for super high resolution projects, this could take days! Even with a managed render farm.

Tiles-And-Splits-02.png

Tiles-And-Splits-03.png

Tiles-And-Splits-04a.png

Save everyone some time and do it on your tricked out laptop

Multi-Screen Playback

Alright doggy - so you want to have multiple screens in sync? We know that we can get 7 @ 4k30 out of 16-Core i9 with 2x RTXA4000 cards. We know we can get the same out of an 8-Core i9 with 2x P4000 cards, but the performance sometimes dips. 2x Titan X cards also work, but it struggles a bit, too. Curious about what works? DIRECT Playback and mapping through MadMapper or playback through TOUCH and NDI'd/spouted to Mad for mapping.

Here are some build breakdowns and their achieved performance using MadMapper.

ID	CPU	GPU	RAM	Display Outputs (channels @ rez @ fps)
Krispy	Intel i9 8-Core / 16 Threads	2x P4000	64GB	7 @ 4k @ 30fps +1 Control @ 1080
Donna	Intel i9 16-Core / 24 Threads	2x RTXA4000	64GB	7 @ 4k @ 60fps +1 Control @ 1080
4x	AMD 24-Core / 48 Threads	4x RTXA4000	128 GB	10 @ 4k @ 60fps +1 Control @ 1080 (could probably handle more)

Here's a thread on MadMapper's forums describing the best workflow - again - who the F is this Trashcan guy??? He seems really smart.

FWiW - these builds are capable of channels x4 of 1080 in perfect sync if you split the 4k with something like an FX4 or HA5.

Breakout Example TouchDesigner @ 8 Outputs

6 @ 4k and 2 @ 1080p + Console Display

A summary of emails sent around by Sean Leo, Raphael Palefsky-Smith, Dave Tennent, Cam Vokey

The Good News

You can get 7 channels of 4k @ 60fps out of a 2x GPU system if the cards are P4000 or greater (newer flavor is the RTX A4000). Why 7 channels when there's 8 outputs? You need a control monitor for your console, and if you use the on-board output, it drags your overall performance down. Technically this config can give you 28 displays with 7x fx4s (or HA5-4ks) – really cool. Obviously, you can draw headless (without a console) and have more screens, but I've found that if you can have a dedicated console, you should! Saves some pain down the road.

Mixed resolutions aren't your friend, but your total res is actually not bad. My recommendation is to check your performance and then combine your approaches so your computer is looking at only 4k displays. Your 6x 4k outputs are native through the GPU and your 2x 1080 outputs are through your FX in output #7 of the computer. Save output 8 for your console. Don't use the onboard video out (if you have one), it'll drag your performance down.

Performance Testing

First – you should find out if your computer is capable at playing at full resolution without being connected to external displays (make a TD window that's your target resolution total knowing you'll only see a portion of it). Watch your frame rate. If there's struggle here, your problem is your computer spec or your fx are too heavy. Or perhaps your level of TD license.

You can try pre-rendering some content at that resolution and playing it out using TD or MadMapper. In general, playback performance in MM is better than TD (and you get less tearing, etc). Also, MM is rentable.

What kind of content : render at your target resolution and embed some sort of timecode reference so you can visually see if you're dropping frames. Something that flashes every other frame is a good representation of performance drag.

You can also put a timecode number on every screen to check sync. Take a photo with your camera, if the number is the same (or even just between frames when you've got displays with different refresh rates), you're probably good.

If you have a TD Pro license, there's a special sync sauce at that license level. I've been in situations driving tons of 1080 displays with a comm license running – I plug in the Pro license, and the playback is suddenly perfect. If the media works in Mad, but not in TD, this should fix that problem. I'm not sure what the gatekeep feature here is, but it's pretty wild. Just checked and it seems you still can't rent a license :(

Once you get base performance to a level that meets expectations, read on.

GPU Outputs + FX4 Things

PCs (and Macs), while capable of playing out to a ton of displays, GPU bottleneck is created in a ton of different places, but the lowest lift is setting your computer to thinking it's using less displays than you actually are. The Datapath FX4 does this great as an external device, but your output resolution will be limited to 1080 per screen. If you're ok with that scale down on rez, then this is 100% a good call.

If you do explore the Datapath route, you may want to look into a sync generator and getting a sync card for the PC. This would help lock time between your GPU's, TD and the Datapaths.

Some GPUs allow you to create a FX4 like capabilities with the GPU config. On NVIDIA (non-gaming cards), this is called Mosaic. You can create a display that's 7860x2160, for example. This will

improve performance massively – drawing the same exact resolution to less displays, even though the rez is the same, is more efficient!

TouchDesigner Things

If using TD with Mosaic, you should look at GPU Affinity, which binds a TD process to a specific graphics card. The Derivative site has a good write up on the pros and cons of this method.. If you go that route, displays need to be matched, so use an FX4 for the two 1920 displays and a control monitor so the graphics cards are only looking at 4K displays.

One note from the TD programming side: The website recommends that you split the TD patches so one patch handles displays 1-4, patch 2 handles displays 3-7, and then both patches need to be launched via a BAT script. This would have some implications if you're doing content that wants to wipe across the entire canvas but that depends on the brief.

Other Methods to Improve Performance

Sometimes, sending media via Spout/Syphon or NDI from TD to MadMapper and then drawing to your displays there may actually improve performance. This is counter-intuitive : you're doing more work. Mad is just really efficient at drawing to displays. Also, it's a good place to crop and scale. If you need to add margins in between content here, or need to zhuzh some positions Mad is awesome at that.

Sometimes, having different display EDIDs brings performance down. Some headless passthrough (they fake an EDID) at 4k in all of your outputs may help.

Also you can try to reduce what your computer is doing – what is pre-render-able? What can be achieved through other means? Bake as much as the playback content as you can which will limit the amount of operations in TD, which will buy you some more overhead for performance.

Codec Stuff

The performance between NotchLC (NLC) and HAP on a TD system was fairly matched. The move was to pick the codec based on image quality. NLC was a bit better than OG HAP or HAPQ. HAPr (a newer proprietary flavor was recently improved by Jokyo) was better at gradients than NLC. The original HAP code included a version of HAPr (thanks Blair). But don't think you can encode / decode HAPr currently with FFMPEG.

You Can Also Try

Splitting the content across multiple systems and syncing via LAN. I avoid this as much as possible, but there are ways.

Additional Thoughts

TouchDesigner recently added the Direct Display Out TOP - a bit tricky to set up and you need Windows 11 IoT/Enterprise and a Quadro card, but it's really nifty. Instead of configuring a bunch of OS windows, placing them on the right monitor, hiding the mouse, worrying about DPI... you just send a texture directly to the monitor. It might make the setup harder for future maintainers - the monitors stop showing up in the Windows display settings, kinda confusing - but worth it? Sounds like it tricks your NVIDIA GPU into functioning like a BMD card.

Also: looking forward to the new Blackmagic 2110 card, which can send out 8x 4K streams or 32x HD over ST2110 Ethernet. So many pixels! You'd need a beefy 100G switch and a converter for each monitor. ST2110 is getting more and more traction.

Bottom Line

This resolution out of a properly configured single system (RAM, GPU, CPU etc) will work as long as the live rendering isn’t too complex. Find out what you can definitely achieve and then slowly add all your features back in until you get to your sweet spot!