Nucleus by Roon Labs

Introduction

The Nucleus and Nucleus+ appliances were conceived to bring Roon to a wide audience of music lovers, particularly those who aren't highly technical or interested in do-it-yourself (DIY) projects. By tightly integrating high-performance hardware with an optimized operating system and purpose-built software, they provide a simple way to get the richest possible Roon experience. They represent the culmination of an extensive product design and engineering effort, combining the Roon team's years of experience designing media server hardware and extensive software development expertise.

The Nucleus product line was developed with several goals in mind:

- A turn-key Roon *Core* that does not require a Mac, PC, or NAS
- Computing power to support Roon's requirements now and in the future
- Ease of use – little to no customer support required after installation
- Software and firmware updates downloaded "over the air" and managed by the end user
- Reliable and robust operation – tamper-resistant and nothing to service
- Audiophile-friendly – no fans or moving parts

This document provides insight into the design and engineering choices that were made during the development of the Nucleus product line in order to accomplish these goals.

Product Specification

Before choosing a hardware platform or deciding on a software strategy, we set out to specify the features that we thought would be most important to Roon users. During this process, we discovered that there is rarely a single, universally "correct" way to address a feature, and many tough choices were made along the way.

HORSEPOWER

It was tempting to try to design a single device that was perfect for everyone. But since its launch in 2015, Roon has evolved to include a variety of computationally intensive features like DSD upsampling, convolution, and support for music libraries in the hundreds of thousands of tracks; specifying hardware to meet the needs of the few people who use *all* those features would have driven the price of Nucleus up for everyone.

We made the decision to produce two models, identical in every way except raw computing horsepower: **Nucleus** for the majority of users, and **Nucleus+** for those with the most rigorous requirements in library size, numbers of simultaneous playback streams, and high-rate DSD.

MEDIA STORAGE

When the first media servers appeared in the 1990s, storing large media files was their primary application. Large hard drives didn't exist, consumer NAS appliances were not yet common, and the network speed (both LAN and WAN) required to stream high-resolution media content was extremely expensive.

The world has changed since then, as have the requirements for media storage. Some people store their music on a single USB disk, while those with large music libraries often have a NAS with RAID for redundancy and fault tolerance. Audiophiles may prefer SSD (solid state drives) over traditional spinning hard drives because they have no moving parts (and thus no mechanical noise) and a longer life span. A new generation of music listeners is increasingly relying on cloud-based streamed content rather than local media files, so they have little to no need for local storage.

Nucleus aims to address all these cases by taking advantage of modern storage technologies, most notably the introduction of low-power, high-capacity 2.5" hard drives and SSDs. Nucleus has an internal 2.5" drive tray, which can accommodate a single SATA SSD or HDD for dedicated internal storage. Two USB 3.0 ports can be used to connect one or more external hard drives with support for a variety of common file system formats: FAT32, NTFS, exFAT, and HFS+ ("Mac OS Extended"). Of course Roon has always supported connections to SMB/CIFS file servers, which includes most NAS appliances available today.

NETWORKING

Wireless networking is ubiquitous, and often consumers expect that every device can and should be wireless. While Roon is often controlled over WiFi (using Roon Remote on a mobile device) and RAAT (the protocol used by *Roon Ready* devices) is designed to work reliably over WiFi, Nucleus is an infrastructure product. Because its primary function is to act as the Roon *Core* on a home network, reliability is a key networking requirement. We made the decision not to implement WiFi in Nucleus, and rather to require a connection to a router or switch via Gigabit ethernet.

OPTICAL MEDIA

For users of early media servers, ripping CDs was a primary use case for customers. Our research indicates that these days, the majority of people have already ripped their CD collections. Among those who continue to buy and rip discs, there is often a strong preference for a particular ripping application (like EAC, dbPowerAmp, or XLD) which are only available for specific operating systems.

We also know from our experience that optical drives are inherently prone to failure over time. Rather than building in a CD drive and implementing our own ripping software, we decided to leave

ripping in the domain of the PC. Media files from a computer can easily be copied to Roon by SMB transfer or via drag-and-drop to the Roon application.

AUDIO OUTPUTS

Almost all music "server" or "streamer" products in the market have audio outputs of one kind or another, either digital or analog using a built-in DAC (digital to analog converter). With Nucleus, we have taken a different approach.

A cornerstone of the Roon product strategy has always been to support a wide variety of partner audio brands. Roon's role is to be the best possible music source, leaving rendering to products from companies that specialize in audio. Nucleus extends that strategy by having no analog audio outputs at all. External DACs, receivers, integrated amplifiers, and speakers can be connected either directly via USB, HDMI, or over the network using RAAT, AirPlay, or Roon's other supported protocols.

POWER SUPPLY

The Nucleus requires a 19V DC supply, which meant we had several options. The most elegant product choice was designing an internal AC-DC switching power supply board, which would have meant a single external AC power cable from the Nucleus to the wall. We rejected this approach, though, because we realized that in our target markets, one size doesn't necessarily fit all.

Depending on whether the Nucleus is connected to audio equipment directly (via USB/HDMI) or indirectly (via ethernet) there could be advantages to different switching power supply designs, and the possibility of a DC filter might make sense in some configurations. In addition to those options, in the audiophile world there might be a preference for a linear power supply, which would require a different thermal design, substantial additional cost, and more safety certifications.

Taking all these variables into account, we took the pragmatic decision to include a simple external "wall wart" DC power supply with the Nucleus. The wall wart is an appropriate solution for the vast majority of users, and for those with different requirements, a wide variety of third party switching and linear supplies and filters are available.

COOLING

Almost all modern computing hardware includes a fan-based thermal solution for maintaining internal component temperatures within safe limits. Unfortunately, fans are inappropriate for critical listening environments, so a completely fanless cooling system was an absolute requirement.

DOCUMENTATION

It's customary for audio components to ship with extensive printed documentation, but in the world of computing and IoT devices – whose features are software-driven – product documentation is fluid and constantly changing. We took a cue from that world and included only a simple Quick Start Guide

with the Nucleus; the more detailed User Manual (and other resources like Knowledge Base and Community support) are available online.

Hardware design

Roon is a demanding piece of software, and it doesn't live up to its full potential on underpowered computing hardware. To operate at its best, it requires a modern Intel Core i3, i5, or i7 CPU, at least 4GB of RAM, and an SSD for storing the operating system and databases (spinning disks are fine for music storage).

Prior to the development of Nucleus, we explored collaborations with hardware manufacturers in which support for the Roon *Core* would be added to their hardware products. While a few of those projects did come to fruition, most did not because too few hardware platforms were able to provide the computing power required to run Roon.

We encountered a second problem as well: Typical product development cycles in audio are poorly matched to the rapid development pace of a dynamic software product like Roon. Modern computing platforms have a lifespan of two to three years, and modern software aims to take advantage of that pace. In the audio world, a decade or more might pass between a component being chosen and the last product using that component being sold to a customer.

PLATFORM: THE INTEL NUC

The optimal hardware platform for Roon is one which provides a small number of high-speed CPU cores. To provide the best possible user experience, Roon is designed to take advantage of a high-speed dual-core CPU, fast RAM, and a fast SSD.

Many media servers and other connected devices use low-power embedded hardware platforms designed for long product life (up to ten years in some cases). These platforms have the advantage of being reliable and cost-effective, but they're often supplied by small manufacturers with limited software engineering resources, so the driver updates and security patches required to keep a modern Linux operating system up to date are unavailable.

An alternative is to use widely available consumer platforms like those found in modern PCs. The cost of those platforms is low and software support is generally very good, but product cycles are as short as six months in many cases. That means that physical form factor can change from one product cycle to the next, and it's often necessary to build a new operating system kernel for each generation of the platform, which is costly in terms of development, testing, and maintenance.

With Nucleus, we have taken a completely different (and somewhat unconventional) approach. Rather than using an embedded or consumer platform, or investing resources in a custom motherboard, we've partnered with Intel to build Nucleus around their NUC platform. This allows us to leverage a stable and well-supported family of computing components that behave consistently

from one generation to the next, share a form factor, and are released on a regular schedule within a predictable price structure.

Working with a company that has Intel's resources, commercial focus, and engineering capability has been invaluable for us. We've had the opportunity to collaborate with their engineering teams on topics like power management and thermal design in order to deliver an extremely impressive amount of computing power in a compact, fanless appliance.

Intel's NUC platform gives us stability while future-proofing our development efforts, due in large part to the consistency in form factor, BIOS, chipsets, and board layout across the product generations. It also means that every revision of Nucleus takes advantage of the latest and greatest hardware Intel has to offer. For example, most of the development work for Nucleus took place using 5th and 6th generation NUC hardware, while the first production Nucleus units use a 7th generation NUC. Had we chosen an embedded platform at the traditional point in the product development process, Nucleus would have been two years out of date by the time it arrived on store shelves.

Using the NUC allows us to focus our investment on aspects of the product that will be longer-lived than a single hardware revision: industrial design, thermal management, mechanical engineering, Roon OS, and the Roon software itself. Whenever newer, faster NUCs become available, we'll be able to update the Nucleus product line with minimal additional design and development effort.

MECHANICAL AND THERMAL DESIGN

Because the Nucleus will likely be located in critical listening environments, managing heat and power without fans was a primary concern during the design process. Beyond protecting the internal components, the function of the chassis is to act as a heat sink to spread and passively dissipate the heat produced by the CPU. Thus the mechanical and industrial design were informed by a balance of thermal, aesthetic, and commercial considerations.

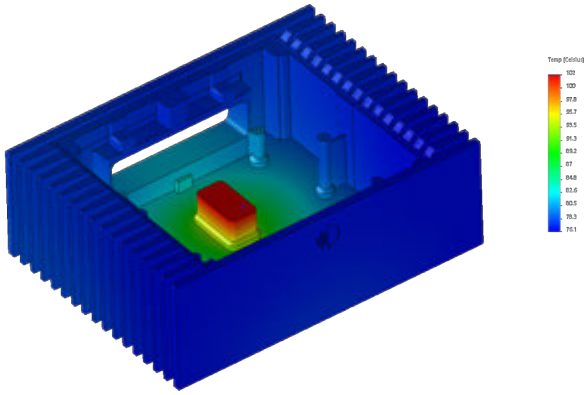
A typical consumer electronics device is assembled from numerous CNC (machined) and extruded aluminum components. The tools required to produce these types of parts are relatively cheap and quick to produce, and unit production costs are low. The thermal mass of extruded aluminum is also fairly high, making it a good choice for producing low-cost, efficient heat sinks.

The challenge with Nucleus was that our ambition for an elegant industrial design precluded the use of extrusions. We were after a monolithic form without seams or exposed assembly hardware, and since the extrusion process produces parts with a continuous profile (think toothpaste being squeezed out of a tube) those parts can only contain geometry in two dimensions. Our industrial design *(see below)* required detail that typical CNC processes could not reproduce, so machining from solid aluminum was also not an option.

The analysis came down to a choice between a "do it fast" and a "do it right" approach. Given the constraints of the available manufacturing techniques, we decided on a one-piece die-cast aluminum chassis. Die casting is a simple process, but developing a tool is costly and time consuming; the

aluminum used for casting is also more porous (with a lower thermal mass), so a cast chassis requires a higher volume of material to achieve the same thermal efficiency as an extruded component.

Using flow analysis, we modeled the heat flux of the cast chassis and designed a “mesa” to act in place of the fan and shroud included with the stock Intel NUC. The dimensions of the mesa provide the largest envelope to the CPU surface that doesn’t impact the other components surface-mounted on the motherboard.



The thermal characteristics of the design were carefully tuned in simulation during the engineering process.

Mechanically, the main goal of the design was simplicity. There are only two internal PCBs (printed circuit boards) aside from the NUC: one to house the power switch and LED indicator, and another to provide a bridge to the SATA connector for an optional internal SSD or hard drive. The machined aluminum bottom plate forms a user-accessible hatch to install the drive.

INDUSTRIAL DESIGN

The industrial design of the Nucleus, a physical manifestation of the Roon *Core*, aims to be bold and simple, functional and durable, yet visually striking to suggest the richness of the Roon experience that it enables. Kubrick's *2001* monolith and the puzzle box from the *Hellraiser* franchise were commonly referenced themes in design discussions.

The case is realized as a single block of die cast aluminum with no visible seams or markings except for the subtle embossed Roon icon on the otherwise flat front face. In keeping with Roon's ideology, the Nucleus' form is its function – it is a continuous heat sink in the round, wrapping the masked internal volume so the profile of the “fins” is never fully apparent. The boundary of the object appears to shift from solid to diaphanous depending on the perspective from which it is viewed.

The finish of the Nucleus is a multi-layered system typically applied in the automotive industry; satin, colorless, and a value just shy of black. The surface disappears in shadow and picks up subtle highlights to express the geometry of the Nucleus. The only LED indicator is indirect, casting subtle light on the power button which is located on the back of the case. This placement is deliberate and

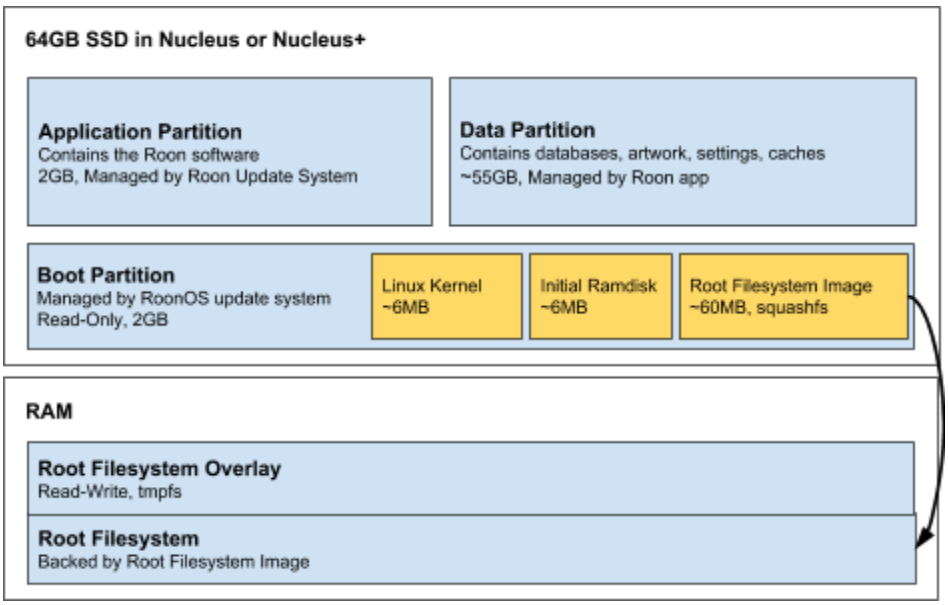
obviates unwanted visual disruption in the most sensitive spaces. Back panel ports and switches are set back unobtrusively in recesses inspired by the hangar bays in the Death Star from *Star Wars*.

Roon OS

The Nucleus and Nucleus+ utilize a custom operating system called Roon OS, a Linux-based operating system built from the ground up, designed and tuned for use in media appliances. Roon OS is the heart of the Nucleus, and the source of most of the technical innovations in the product.

COMPARTMENTALIZED FILESYSTEM LAYOUT

Most media server products utilize a single read-write filesystem that contains the operating system, user-specific databases (including settings and caches) and application components. Roon OS is built with a compartmentalized file system layout in order to provide increased robustness against user error, hardware failure, and software bugs.



Solving the "full hard drive" Problem

For any device that stores user data, full hard drives are a frequent cause of support incidents. This problem often results in confusing symptoms and lengthy troubleshooting exercises.

Roon OS attacks this problem in two ways: by reducing the likelihood that this happens in the first place, and by ensuring that if it does occur, the problem is compartmentalized so that the device does not end up "bricked" (unable to boot).

All three partitions – Boot, Application, and Data – are separate from one another and managed with different access policies. A "full drive" situation in the Data or Application partition cannot stop Roon OS from booting, and in most cases a full Data partition does not stop the Roon *Core* application from

starting. The crucial Boot partition is mounted in read-only mode and only written to by a small piece of carefully audited code that is responsible for updating Roon OS.

Solving the "data corruption" problem

Data can become corrupt as a result of user error or software bugs. The compartmentalized design of *Roon OS* limits the impact of such incidents.

Most cases of corruption occur in the Data partition. This is because the partition stores user data, which varies in size and content from user to user, and also because users are able to access this partition directly as a network share.

As long as the Nucleus is bootable, it's possible to access the management interface and restore the product to normal operation. Since Roon allows the backup of user databases to local drives, network shares, and Dropbox, a Nucleus can always be restored to its last working state with minimal effort.

Solving the "enterprising hobbyist" problem

In our experience with high-end audio products, we're frequently surprised by how their design makes tampering or end user modification of the operating system very easy. Some products boot from an SD card that is accessible to the user on the outside of the enclosure. These are easily tampered with by mounting the card and modifying its contents with a PC. Other products have published or easily-guessed root passwords, often allowing full access over a network.

Nucleus products have neither removable boot media nor a root password. The compartmentalized filesystem model means that on each boot, the Root Filesystem is recreated in a fresh state from a known working image, and on each update it is fully replaced. This "self-healing" quality means that any modifications made by an enterprising user would be destroyed either on the next reboot or the next update. This inconvenience is a significant deterrent to hobbyists interested in tweaking the system, and ensures that the state of a Roon OS can be trusted throughout the life of the product.

We plan to provide a mechanism for running third party Extensions on Roon OS in the future. This will be done in a secure and sandboxed manner that does not expose Roon OS to third party code directly. This way there is no chance of damage, and no chance that an accidental dependency on Roon OS implementation details cause extensions to break over time.

UPDATE SYSTEM

Roon OS updates are downloaded from the internet and confirmed by the user via the Roon app. Dealers and installers are not involved in the update process, and no physical access to the hardware is required.

Because Roon OS is so compact, it can update or reinstall itself in just a few seconds after downloading approximately 50 megabytes of data. Typical media server operating systems weigh in at hundreds of megabytes to a gigabyte in size.

Each update fully re-installs Roon OS, including the kernel. This guarantees that devices running the same OS version are always in the same state. Updates are installed *atomically*, meaning that it's safe to power off the device or reboot during an update without the risk of corruption or a partial install.

Roon OS accomplishes this by installing the new version of the operating system side-by-side with the previous version, verifying that the new version is intact, then finally atomically modifying a single location on the SSD to point to the updated version. Only after the Nucleus has rebooted successfully using the updated operating system does it remove the previous version.

REMOTE SUPPORT

Roon OS includes a mechanism that allows the Roon team to remotely access Roon OS devices for diagnostic and support purposes, both proactively and reactively.

The remote support mechanism is built on top of Roon's "push" infrastructure. Similar to systems deployed by smartphone and tablet vendors, our push infrastructure provides a mechanism for initiating communication with devices even when they are located behind a router or firewall.

BUILDING IT THE RIGHT WAY

Most embedded Linux operating systems in the audio world are built in one of two ways: either a desktop- or server-oriented Linux distribution like Fedora or Debian is trimmed down to "fit" on an appliance, or an "embedded" operating system is built in a semi-automated fashion using a system building tool like yocto or buildroot.

Wrong way #1: Trimming a desktop or server Linux distribution

Linux distributions designed for desktop workstations or servers are easy and quick to get up and running, but are not appropriate for embedded devices. Minimal installations of these distributions start at hundreds of megabytes and typical products built using this approach have an operating system that is over a gigabyte in size. This is both inefficient and unnecessary.

Furthermore, the software update process for these operating systems is based on package managers that are neither atomic, repeatable, nor fast. Hundreds or thousands of packages are updated one at a time, modifying the installed operating system in place during the process. Because of their complexity, these package managers must often be attended by a technically skilled system administrator. Reboots or power interruptions during updates of this kind can cause corruption to the operating system and result in a "bricked" device – one which can no longer boot.

Products built this way tend to fall out of date over time, since performing major operating system upgrades in the field is risky. Sometimes, the operating systems end up so far out of date that security

updates are no longer available, leaving users vulnerable. When a major operating system upgrade becomes critical, manufacturers often fall back on primitive update mechanisms like memory card swaps, which fail to reach the whole user base.

Wrong way #2: Using an "easy" system builder

"Easy" system building tools like yocto and buildroot appear to solve these issues. They do indeed dispense with package managers and make it easier to create a smaller operating system (generally hundreds of megabytes instead of gigabytes) but they have some issues of their own.

These tools hide what should be a thoughtful, considered system design process behind an interface whose primary goal is to make system building easy. Unfortunately, the main reason that "easy" is important is because it is common for companies to press hardware engineers into creating an embedded operating system without Linux or software development experience.

Too much power and control is lost by making it "easy." System building requires a detailed understanding of the technologies, careful decision making, and attention to trimming bloat.

A better way

Roon OS departs from both of these approaches. The operating system is assembled using a proprietary build system developed at Roon Labs. Each package that makes up Roon OS was selected and configured by hand, carefully considering how it will be used and making significant effort to keep the operating system as trim as possible, both while the system is running and at update time.

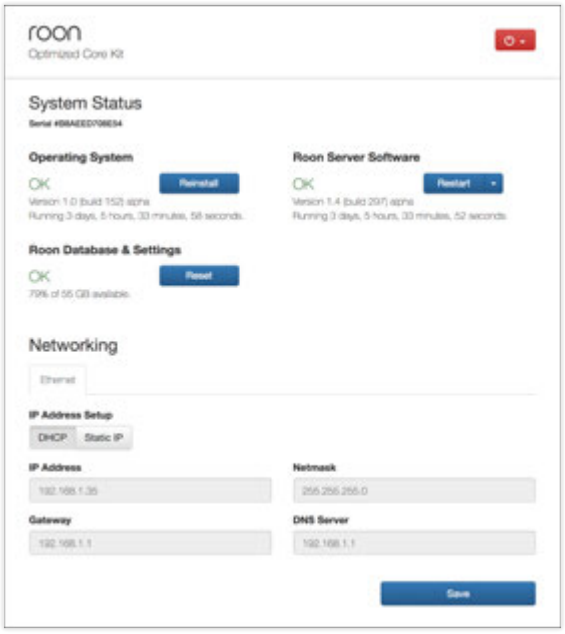
This custom approach is what enables Roon OS's compartmentalized storage model, fast and atomic update system, quick boot, extremely small size, and runtime efficiency. It also means that our engineers fully understand the components that make up Roon OS end to end, and that we are in a position to keep things up to date or fix problems in any area of the system.

SELF-SERVICE MANAGEMENT INTERFACE

The Roon OS management interface is a crucial ingredient in making Nucleus easy to support.

Many products have web interfaces for configuration or status reporting, but the Nucleus web interface has some extra capabilities. From the Roon OS web interface, it is possible to:

- Re-install Roon OS
- Re-install the Roon Core software
- Reset data on the device to a factory-fresh state



These options allow novice users to recover from major issues or put the device into a safe state with minimal guidance and without physical access to the hardware.

NON-NUCLEUS APPLICATIONS

Roon OS is also available to hardware partners interested in building their own NUC-based server products that incorporate Roon Core functionality. For end-users, Roon publishes a DIY oriented product based on Roon OS called “Roon Optimized Core Kit”, or ROCK, which allows users to build their own media servers using their NUC of choice.

Conclusion

Together, Nucleus and Roon OS represent a leap forward for media server appliances. As a result of thoughtful product design, hardware engineering, and software development, they provide a turn-key solution for installing and running a Roon *Core* that is easy to support, high-performance, and robust.