DFI

Flexible deployments in industrial and commercial operating environments by MicroK8s with DFI EC70A-KU/GHF51

DFI's EC70A-KU and GHF51, both Ubuntu-certified products, can easily create and manage peripheral nodes through MicroK8s and related packages painlessly and without the burdens of compatibility. It can complete automatic deployment, expansion, and execution of application containers across host clusters, while also reducing application maintenance costs to extremely low levels.

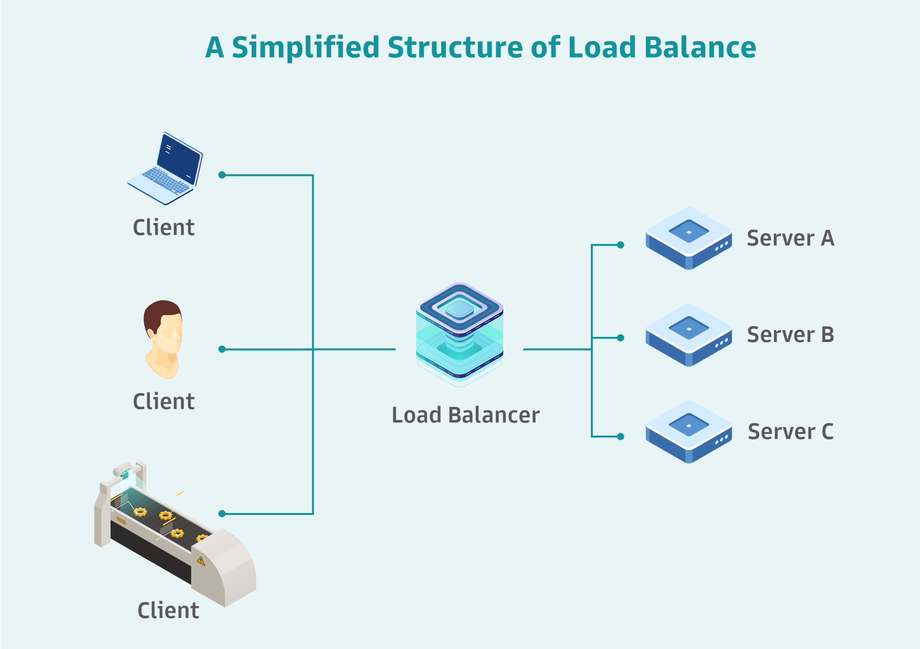
Industry: **Smart Retailes, Factory Automation, AIoT**
Application: **IoT Node Deployment, Load Balancing, and Redundant**
Solution: **EC70A-KU and GHF51**

Case Study: Application Note

With AIoT devices in industrial and commercial applications gaining popularity, the Internet of Everything has significantly improved the efficiency of production and operations. Big data collected from numerous edge devices has also allowed for the policies in implementation to be continuously revised, and for the cloud and terminal to communicate smoothly, forming a positive cycle of efficiency.

In addition, in recent years, various fields have showed urgent demand for unattended operation methods. Accordingly, many companies start to actively establish their own private clouds to achieve a high degree of management flexibility through remote control. Quickly building and deploying terminal devices and conducting unified management in the private cloud have become a mainstream strategy in cross-applications. Load balancing and high reliability are the two foundations for maintaining the efficiency of private cloud operations as well as basic indicators for evaluating the maturity of production and business models. High reliability is defined as the standard of the online ratio of services or productivity within a certain period of time. Load balancing is a common method for maintaining high reliability. It aims to share the loads among devices and provide support when other devices fail; that is, it "catches the ball" to avoid "dropping it."

Whether it's a convenience store kiosk or production equipment in a factory, not achieving load balancing will likely lead to services or productivity being interrupted due to hardware or software factors. This represents a decrease in reliability, which will cause loss of profit or costs for downstream manufacturers; as for upstream manufacturers, they lose the foundation for customers' trust in their solutions.



This is a brief illustration of the load balancing mechanism. The left side is the demand side, the right side is the processing side, and the middle is the load balancer. The load balancer must adjust and allocate resources according to load conditions to ensure uninterrupted services.

Achieve Load Balancing and High Reliability Through Virtualization

We are in an era that relies heavily on hardware virtualization. This trend will continue. Abundance of hardware performance makes it possible to deploy multiple containers in a single device. If each container represents a virtualized service entity, such as package delivery at a convenience store, even if the software in each container crashes, the system can immediately call another container's service to take over. A container can be dispersed in multiple devices. Another application example: A Panel PC is used to display flights in the flight display of an airport terminal. This Panel PC will request information from the database host responsible for the integration of flight information. If the database host is powered off or if a hardware error occurs, the Panel PC will access a backup database host so that the display of flight information is not interrupted.

Load balancing is especially common in factory automation. Suppose that there are 10 cameras in a production line used for defect detection and that there are 2 edge servers for AI inference on the backend. If one of the servers already bears the data volume from 5 cameras, when the 6th camera goes online, the system can be assigned to the second server that does not yet have a load or only has a low load. Avoiding an overly heavy load on the first server and adversely affecting the performance of AI inference.

That sounds very ideal? In the past, however, manual deployment of applications into containers and cross-device management were tedious tasks. Engineers had to add or remove containers on each device. Also, there was no automatic backup function or automatic expansion function among containers.

Think of the cash register in a retail store as an analogy. In most Asian countries, convenience stores operate 24 hours a day, meaning that the cash register system has almost no downtime. Have you ever encountered a cash register software update while you were checking out? Services like this with almost zero downtime must rely on rolling updates to keep software up-to-date. Suppose there are 20 cash registers in a hypermarket; 5 of every registers are connected to a host at the back end to obtain price data. Suddenly, during the Black Friday consumer surge, it is discovered that there is a bug that needs to be dealt with. The back-end host can first be updated with some containers from the host while some containers are still operating, so as to minimize risk and impact on operations.

Advanced container management functionality has an automatic expansion mechanism that can automatically adjust the optimal load according to number, capacity, and performance of devices. This must be the most impressive point for engineers who actually use cloud services to develop applications. If we lease five cloud hosts, we can treat these five hosts as a virtual platform by using container management and run IoT services on them, and allow the management framework to automatically determine the load of each device to maximize integration of hardware resources.

The above applications are all extremely simplified examples. In fact, when faced with a large number of IoT devices, it is even easier to achieve consistent integrating (CI) and consistent development/delivery/deployment (CD) through container control to reduce the risk of a small change affecting the whole entity in a large-scale and complex environment.

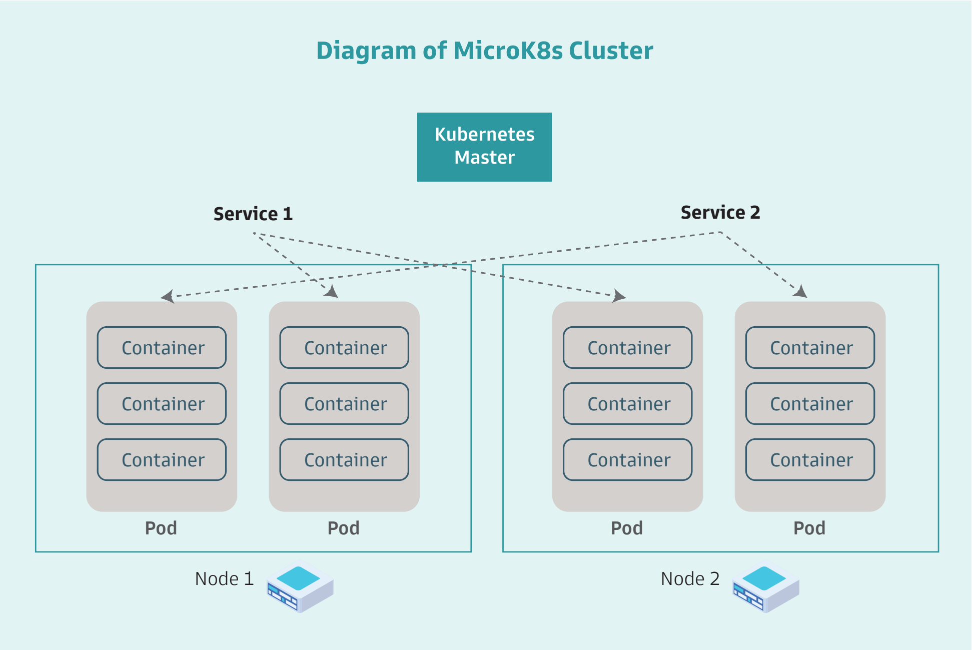
Deploy and Manage Containers Quickly and Easily With MicroK8s

Containers are just indicators of virtualization. What about managing these containers and connecting them into clusters? Kerbernetes (K8s) is the most famous architecture used by many Google cloud services, although it has a certain degree of difficulty and a steep learning curve. At present, the MicroK8s framework on Ubuntu is the top choice that combines practicality and convenience. Of course, each application has its own most applicable framework; they can be easily installed on the Ubuntu system and avoid "overkill". At present, MicroK8s is still the most versatile.

Installation of MicroK8s is extremely simple. The premise is that a Linux system is required. The entire micro container cluster can be established through simple commands. The entire process takes less than a minute, and is very popular in lightweight terminal applications. Coupled with GPU support, machine learning in containers is not a difficult matter.

However, the Linux system driver support for hardware is still a difficult problem that many integration solutions will encounter in design, which in turn makes developers wary of implementation. DFI continues to deepen the cooperation with Canonical vertically, and extends horizontally on products and applications.

The small EC70A-KU system and industrial-grade Pi-GHF51 development board are products that have obtained Ubuntu certification. With no need to worry about support, a lot of time can be saved from debugging. The IoT operating environment can be deployed in the fastest method to provide the



In the MicroK8s architecture, each server can be regarded as a node, the node contains a Pod which is a resource sharing unit, and the Container is the real core running the service. When one Pod drops off, another Pod will be created automatically.

limitless convenience brought by MicroK8s.

The Importance of Passing Ubuntu Certification

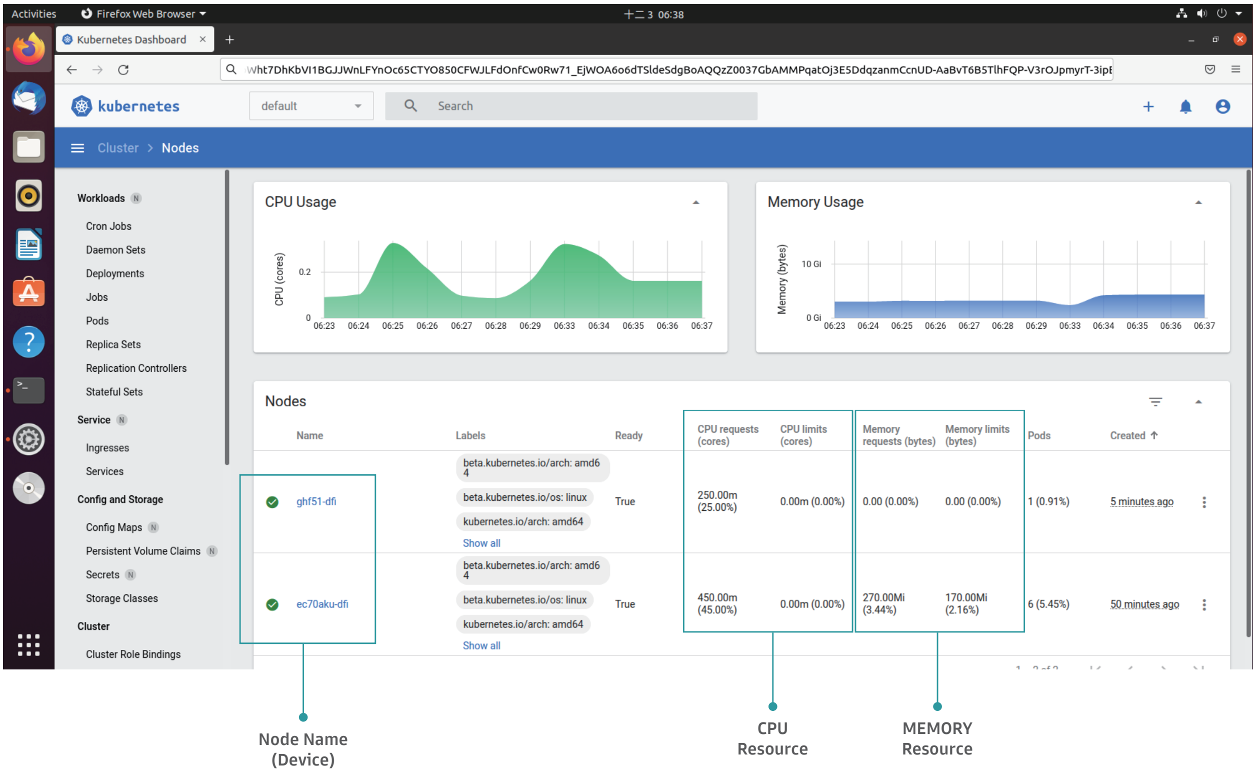
While being safe and reliable, the maintenance cost of the Linux system is also cost-effective, but driver compatibility and dependence on software have always been the perennial problems that are difficult to solve.

Driver problems are not only encountered when installing the operating system, but so long as the operating system is updated, it is likely to happen again, and it is the pain of developers, engineers, and system integrators.

Although there are no operating systems with 100% built-in hardware drivers, and this is even true of Windows, in recent years, Ubuntu's hardware support has improved significantly, and it can now be considered very mature.

Products certified by Ubuntu are free of worry when deploying the operating system to the device and in future updates, and they have priority support. Without a doubt, this is an extra guarantee for the edge application supplier. Hardware products can be introduced with more speed and safety, and the solution development schedule will be more agile.

DFI EC70A-KU and GHF51 have been certified by Canonical and it's 100% compatible with Ubuntu. By using the EC70A-KU and GHF51 to establish operation of the MicroK8s environment, the operator can see the operating status of each node, Pod, and the load of the machine on the management interface of MicroK8s.



Nodes

Name	Labels	Ready	CPU requests (cores)	CPU limits (cores)	Memory requests (bytes)	Memory limits (bytes)	Pods	Created ↑
✓ ghf51-dfi	beta.kubernetes.io/arch: amd64 beta.kubernetes.io/os: linux kubernetes.io/arch: amd64 Show all	True	250.00m (25.00%)	0.00m (0.00%)	0.00 (0.00%)	0.00 (0.00%)	1 (0.91%)	5 minutes ago
✓ ec70aku-dfi	beta.kubernetes.io/arch: amd64 beta.kubernetes.io/os: linux kubernetes.io/arch: amd64 Show all	True	450.00m (45.00%)	0.00m (0.00%)	270.00Mi (3.44%)	170.00Mi (2.16%)	6 (5.45%)	50 minutes ago

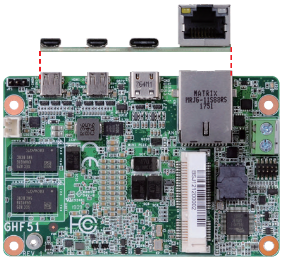
The Smallest x86 Development Board On Earth: GHF51

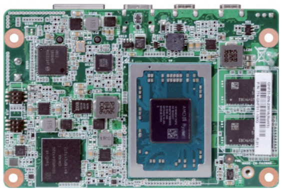
GHF51, known in the industry as “Pi”, is an ultra-small motherboard especially launched for AIoT applications. With an area of only 1.8 inches, it has the same pocketable size as the Raspberry Pi, and is directly compatible with the x86 development environment. This means a wide choice for software packages and that there is no need to worry about porting due to cross-platform issues. Developers can also evaluate performance with higher precision when designing applications.

The industrial-grade design also allows the GHF51 to operate in a wide temperature range. The built-in Mini-PCIe module provides support for Wi-Fi or 5G network, which is perfectly suitable for developing terminal IoT applications.

Spec

- Small form factor 1.8" SBC for space-limited applications
- AMD Ryzen™ Embedded R1000 Series
- Single Channel DDR4 Memory Down up to 4GB/8GB
- HDMI 1.4 resolution supports up to 4096x2160 @ 24Hz
- Expansion and Storage: 1 Mini PCIe, 1 SMBus
- Rich I/O: 1 Intel GbE, 1 USB 3.1 Gen 2, 2 Micro HDMI
- 10-Year CPU Life Cycle Support Until Q1' 30 (Based on AMD Roadmap)





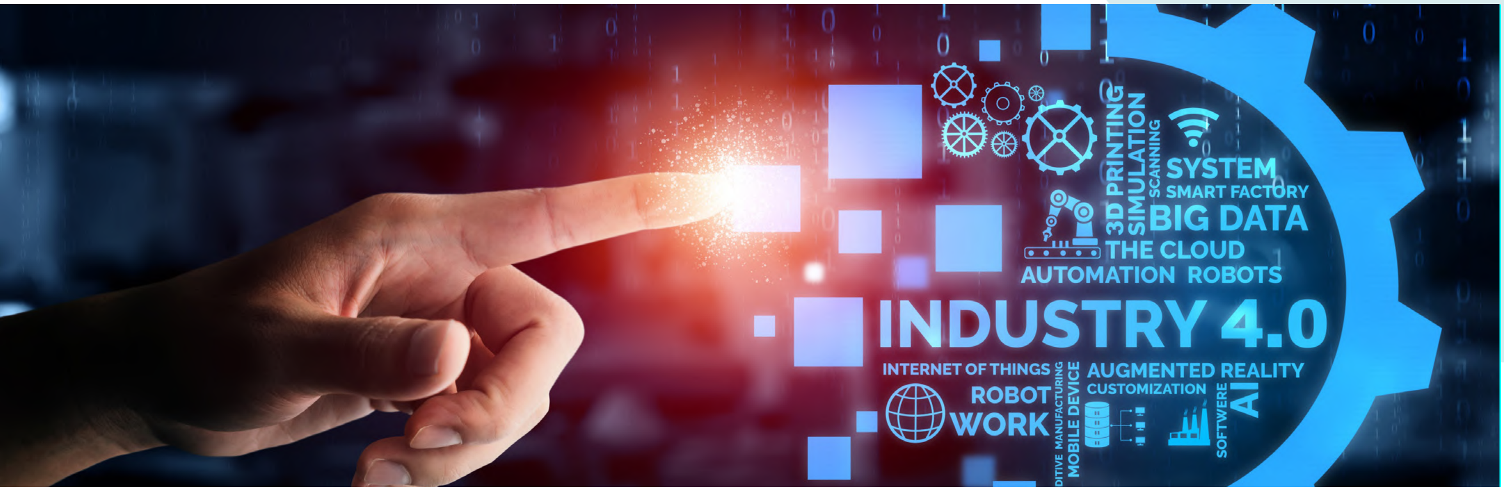
EC70A-KU With Abundant I/O

The ability to link with peripheral devices is one of the biggest differences between industrial-grade computers and consumer products. Industrial-grade production equipment is expensive and has a long service life; also, many specialized pieces of equipment are difficult to update and replace. Therefore, industrial computers need to have a long-life cycle in terms of support for port standards, and traditional or mainstream transmission types must be supported. Serial port support is one such example. In view of this, DFI’s EC70A-KU has made the most compact configuration at the front and rear panels with extremely limited space. There are up to 6 serial ports alone, which can greatly reduce the probability for needing additional expansions in the overall solution.

Spec

- Compact and Fanless Embedded System
- 4GB/8GB DDR4 memory onboard
- 1 x 2.5" SATA 3.0 drive bay (for thickness 7mm only)
- Supports 2 Mini PCIe for wireless application
- Rich I/O ports: 2 Intel GbE, 6 COM, 4 USB 3.0, 4 USB 2.0
- 15-Year CPU Life Cycle Support Until Q1' 32 (Based on Intel IOTG Roadmap)

*The product is not scaled according to the actual size, please refer to the product specification sheet for detailed dimensions.



DFI is proactively and continuously developing products compatible with various certifications, thus accelerating integration and introduction of AIoT solutions

Vertical solution integrators often need to worry about issues such as compatibility and safety regulations, and operating environments must enter into consideration for different countries. From the product design stage, DFI defines the greatest common denominator with requirements that are easily generated in various fields, launches standard products that satisfy the mainstream market, and actively passes and obtains various safety regulations and certifications to help customers integrate and import various applications in the fastest possible time.

DFI also provides highly customized services for customers with special needs. It combines with one-stop ODM to create products that belong solely to you.

If you want to know more, please visit our successful story website.

DFI

Founded in 1981, DFI is a global leading provider of high-performance computing technology across multiple embedded industries. With its innovative design and premium quality management system, DFI's industrial-grade solutions enable customers to optimize their equipment and ensure high reliability, long-term life cycle, and 24/7 durability in a breadth of markets including factory automation, medical, gaming, transportation, smart energy, defense, and intelligent retail.

Website: www.dfi.com
eStore: estore.dfi.com

For more information, please contact your DFI regional sales representative or send us an email: inquiry@dfi.com