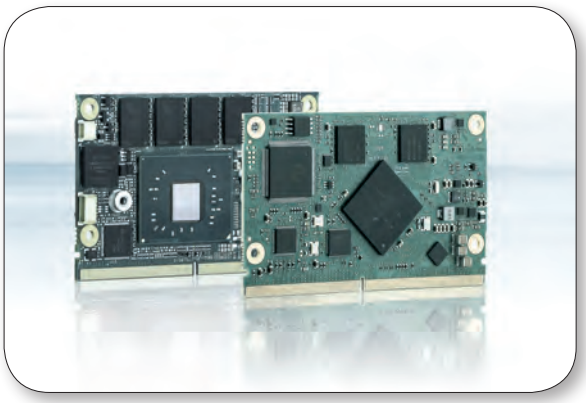
From Edge to Fog to Cloud – IoT Computing with SMARC

By Martin Unverdorben, Kontron

Cloud, Edge and Fog computing is everywhere and everybody is talking about it. In industrial surroundings traditional server approaches cannot provide the required robustness for operation in harsh industrial environments, but how can modular systems based on the tested SMARC Computer-on-Modules provide an efficient solution.



The Industrial Internet of Things (IIoT) is one of the most challenging application spaces to design for, as there are pressures on the developer from clients, management, and the marketplace on price, performance, and functionality. Creating the next generation of intelligent industrial systems will require an elegant juggling act with all three.

There is a trend in IT to bring more web functionality out of the central server farms and inject it into the parts of the infrastructure and the devices operating as close to the user level as possible to reduce network traffic demands, among other things. Data collection and storage is already everywhere, in our pockets and in our homes with personal computers, smartphones, and smart home assistants. It is also the case now in production environments.

Cloud, Fog and Edge levels

Applying IoT technology to industrial systems makes a lot of sense, because we can use and manage the data involved to improve the production process and performance factors such as quality and cost. For example, in a chemical plant the process data is probably already monitored, and some factors, like the temperature and pH values of chemical reactions, can be controlled. Why not store that in data to analyze later? Another example is where products parts are mounted or fixed together with screws, where you store the torque values of each screw to have quality control of all the devices that are mounted there. Peering through the Cloud Figure 1 looks at it from a structural perspective. At the bottom are the devices on the floor of the factory that control the production process directly, which is the area called EDGE computing. On the level above, there are the on-site server racks, which manage the process flow, or control data and monitor and maintain the data stored. The layer on top is the CLOUD, representing all off-premises functionality. The middle layer, which usually already exists in some form in legacy systems, gets new tasks because it must connect to that top-level functionality level, and is now called FOG computing.

So the Cloud is everything that is off the premises, the Fog is the level that's monitoring and controlling, and used to be already there before in the server room in the factory. The lowest level, Edge computing controls the floor and the automation control of motors, sensors, and actuators. These three levels have to function and work together, especially in security and safety, with new tasks to be done and new devices that need to be installed.

3 key functionalities of TSN

Another development coming up is TSN, time sensitive networking. Started in 2012 as an IEEE 802.1 working group, the TSN standard defined a way of networking to ensure very low transmission latency and high availability of all participants. Originally intended to define a network for real-time audio and video streaming, this functionality is perfect to synchronize control of devices on the factory floor.

There are three key functionalities involved. The first is time synchronization, meaning all the devices participating in real-time communication have a common understanding of time. This can also be done with an internal clock, but that can be cost-prohibitive. Using the IEEE 1588 time synchronization eliminates the need for any extra clocks or any extra signals, as the time information is distributed throughout the whole network with this protocol.

The second key functionality is scheduling and traffic shaping, where all devices adhere to the same rules in processing and forwarding communication packets. Already a known concept from the telecom market, it means there are different slices for different traffic classes, allowing you to give a certain packet a different class, or a certain priority. The third is to ensure that all devices comply with the same rules in reserving bandwidths and time slots, possibly utilizing more than one simultaneous pass to achieve fault tolerance. That means TSN is a network that ensures that all partners are talking on the same time level, and also at a very high availability and low transmission latency network.

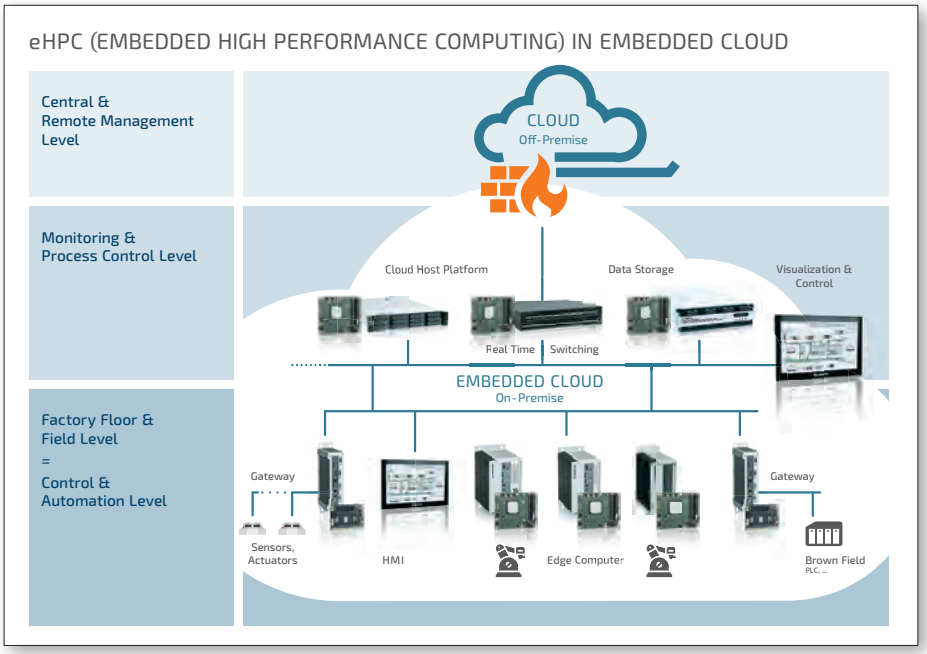


Figure 1. Computing at the Cloud, Fog, and Edge levels.

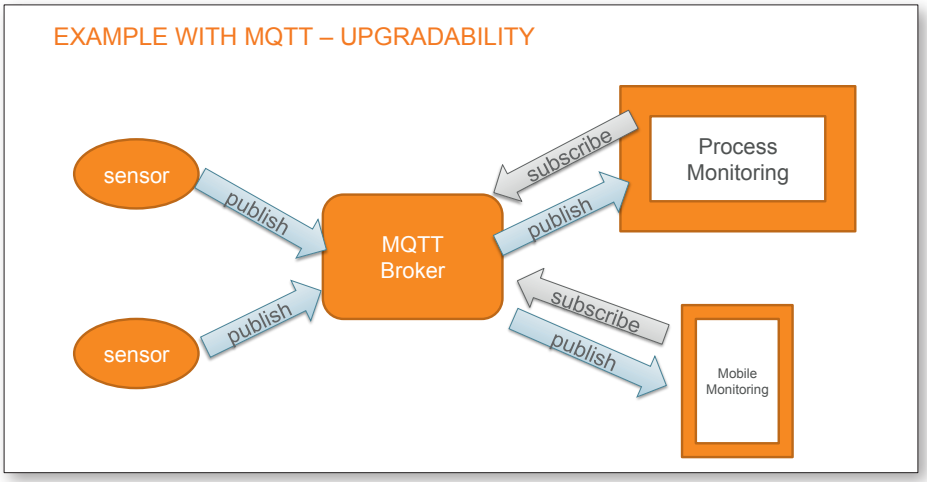


Figure 2. A simple sensor setup connected via ISO standard Message Queuing Telemetry Transport

Security is basic

For every connected device there are three basic rules. The most basic security rule is that every access is authenticated and authorized to that only someone who is allowed to do something can have access. The next rule is that all communication should be encrypted. The last rule, more an awareness, is that all software and firmware can be updated somehow.

But I want to point that out a little bit more in an example that comes from industrial IoT. The example in Figure 2 shows a simple sensor setup, which is connected via ISO standard MQTT, Message Queuing Telemetry Transport, a quite often-used protocol in IoT, located on the application layer, like HTTP, FTP, or DNS on top of TCP/IPN Ethernet. It is a simple subscribe and publish protocol, that allows a sensor, or publisher, to publish its data as a topic. For example, in Figure 2 we have the topic "Factory 1, floor 1, robot 3, oil temperature", which is regularly published by one sensor. If another client is working as a process monitor, it can subscribe to "Factory 1, floor 1, robot 3, #" and then get all that data. So, it's a very simple, but effective, way to track control and coordinate process data.

Now, let's apply the first security rule here, where every access should be authenticated and authorized. For authentication, we need all the participants to be addressed, which means every sensor, every client, and every device needs its own username and password, or its own key file. Authorization is to do what? In this example, the sensor on the top left is only allowed to send to the topic "Factory 1, floor 1, robot 3, oil temperature". Being able to distinguish every detail about who is allowed to do what really makes sense, even when you are in a closed network, as there are intruders that might get access. Authentication and authorization might not be sufficient, especially when there is no encryption in the network. When an unencrypted network client logs in, the credentials are transported in plain text, meaning everybody inside the network can sniff them out very easily. The only thing that you need is a network monitor and access to that network. We can avoid that by encrypting all the transport inside the network. MQTT is really simple because you can set it up on top of any security layer in TCP/IP.

The third rule is that every device's software and firmware can be upgraded. Why is that necessary? Well, let's go back in time, in 2014,when Heartbleed was an issue in OpenSSL, which allowed all the encrypted data be fully revealed to anyone. On a level of 0 to 10, it got an 11. That means all the encryption that we did was simply in vain. It could only be fixed by updating to a fixed version of the software. A second bug came up in 2014 called Poodle. Not as problematic as Heartbleed, but still quite an issue, as it also affected the clients through a fallback from TLS to SSL3, which could be forced by a client, could simply allow a "man in the middle attack". Also, the fix was to update the software, and there is no proof that it will not occur again. Recently we've seen the Spectre/Meltdown issues, which are not as critical, as they only affect machines where already foreign code can be executed.

How to update doesn't matter as long as it will be done, whether locally or remotely. All clients, all servers, all devices that host some firmware, host some software needs to have the ability to be fixed in case of a security problem. For example, when it comes to security, Kontron offers designers secure, trusted boot software to enable a chain of trust to ensure that the BIOS running in the system is authenticated. It is the same on the OS level, with secured operating systems, and there can be an additional level on the application side. All Kontron boards can be equipped with a Wibu Systems security chip with related software to allow full IP protection for running software, where the application code can be encrypted, and therefore, not be reverse engineered. So we can have fully software authorization from BIOS to the application level.

Another use case involves different licensing models. Software can be restricted by runtime, number of program fetches, and other factors, presenting completely new business cases where software as a service can be brought down to the edge layer.

SMARC

When you build an intelligent electronic device, you can go with an out-of-the-box solution, or a full custom design, or something in-between, a modular scalable solution

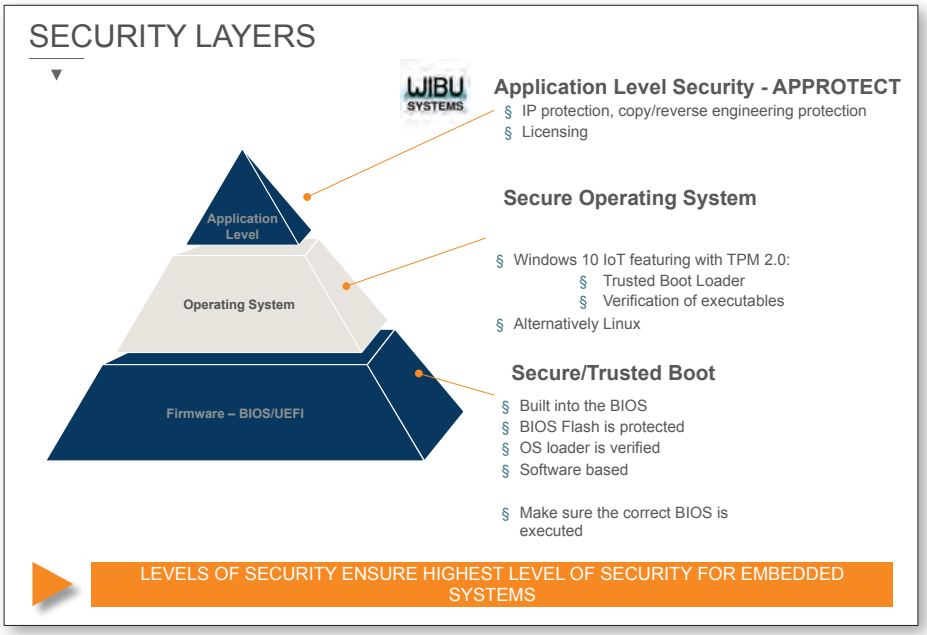


Figure 3. When it comes to security, Kontron offers designers secure, trusted boot software to enable a chain of trust

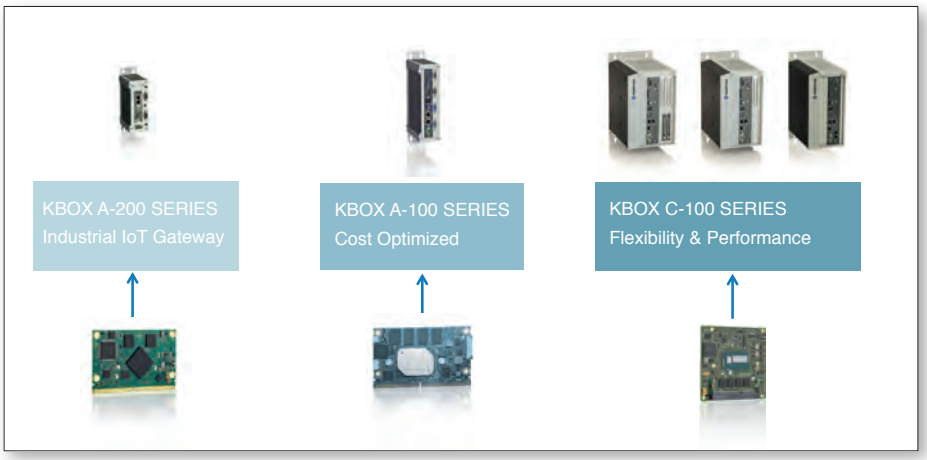


Figure 4. Kontron's KBOX family is a true industrial computer platform, designed to enable predictable productivity in any connected environment

that can be tailored to the application. The out-of-the-box solution is something that you drop in and it works, like a motherboard or video card. If you have some additional requirements, a full custom design is something that you choose when you have very high volumes. A computer and module solution is something that you choose when you have some mid-sized volume, and is a very good compromise between the out-of-the-box solution and the full custom solution.

When it comes to a modular solution you have again two choices. There are proprietary single-vendor computers and modules, or there are standardized computers and modules available from multiple sources. A standard solution offers you a second source so you don't rely on one vendor, letting you scale your devices in performance, power, and price.

Kontron offers the SMARC low-power embedded architecture platform for Computer-on-Modules, based on ARM and X86 technology, to address this issue. SMARC offers a wide range of common computer interfaces, and allows a wide functionality to be implemented. On the graphics side, there is LVDS, HDMI and DisplayPort++, and a camera interface with MIPI-CSI, as well as the typical high-speed interfaces like PCI Express, USB, SATA, and GB LAN, and then all the slower interconnectivity interfaces like audio, I2C, Serial, and such. Compared to other form factors like COM Express and Qseven, SMARC has a few advantages compared to them regarding interfaces. SMARC offers 2 times 1GB LAN interfaces, making it suitable for small gateways that need to address different networks. SMARC also offers four serial COM and two CAN ports, an interface still used in industrial automation.

The SMARC MXM 3.0 connector is currently used in the commercial computer market, so there are many vendors available, and it is proven to be rugged and very resistant to shock and vibration. With 314 pins, it offers more than COM Express Mini, which has 220, and Qseven, which has 230 pins, and the combination of carrier and modules allow a very flat and slim designs.

When it comes to integration, Kontron offers a network card based on PCI Express to bring any computer system directly into a TSN network. SMARC-sXAL is an ATOM® based module for more performance driven applications. There are currently five different CPUs available, the industrial-grade 3E series Atom X5 and X7 boards, and commercial-grade Celeron and Pentium products.

They can connect 1GB to 8GB DDR3L, with ECC support, which is available on the E series. Flash is from 2GB to 64GB in eMMC 5.0, and the graphic connection is fully supported for LVDS, HDMI, and DP++, with triple display support so you can connect three monitors. Then, almost all SMARC 2.0 interfaces here are supported on that module. Operating systems supported include Windows 10, Yocto Linux, and VxWorks.

A good example is the SMARC-sAMX7, the latest available Cortex A7 based ARM SMARC module from Kontron. It is equipped with either a solo or a dual i.MX7 processor from NXP, with an additional Cortex M4 core integrated for small controller applications, where usually an additional microcontroller is used on the carrier.

Another example showing that SMARC is a good choice for industrial IoT is Kontron's KBOX family. There are a variety of PCs intended for IoT, and several are equipped with SMARC modules. The KBOX C series is more for high performance and offers COM Express, and the two KBOX A series incorporate SMARC functionality. If you need ARM-based solutions, then SMARC is the better choice. Also, the height is better on SMARC, so you can have low-profile platforms, and it is better when you need a camera, or when you need a second Ethernet, or when you need CAN.

Looking forward

To summarize, with SMARC you can have the best performance for your IoT applications within the range between ARM and Atom class. It is scalable, enables high connectivity with a wide variety of interfaces, and with Kontron App Protect, you have a security layer to ensure your designs are as safe as they are functional. Properly deployed, this scalable, modular set of solutions can greatly advance your automation system design. ■