RZ/N1 – Multi-Protocol Industrial Ethernet Made Easy

Introduction

With the latest RZ/N1 family, Renesas aims to replace the existing multi-chip solutions in industrial controllers, industrial switches and operator terminals with a single, low-power-consumption SoC (System on Chip).

The high level of integration allows a combination of ARM® Cortex®-A based application core as well as an ARM® Cortex®-M based real-time communication engine cast on a single die.

One of the key benefits is brought by the innovative middleware named Generic Open Abstraction Layer (GOAL) -also called the Operating System Abstraction Layer (OSAL), provided by the industrial Ethernet software specialist the port GmbH. Industrial Ethernet protocol stacks coming from various manufacturers are ported over the same middleware. The benefits are two-fold:

- The change of protocols is simplified in a way that the application does not need to be significantly modified while moving from one protocol to the other.
- The change between the R-IN Engine SoCs is simplified in a way that the application does not need to be significantly modified while migrating from one R-IN SoC to the other.

But let's start from the beginning.

R-IN Communication Engine – innovative support for multiple protocols

The automation industry is moving into a new era often referred as " Industry 4.0". Future production facilities will be based upon cyber physical systems whose key trait is real time and deterministic communication. In order to facilitate this trend, Renesas developed and brought to market the R-IN Communication Engine.

The story of the R-IN Engine started in 2012 when Renesas introduced the R-IN32M3 series of industrial Ethernet communication SoCs which support multiple industrial Ethernet protocols. The IP contains a real-time capable switch, an ARM® Cortex®-M3 CPU as well as various accelerators that take part of the load off of the CPU. There are 4 different accelerators implemented in the R-IN Engine.

- The HW-RTOS is primarily a hardware accelerator that supports the software execution of a real-time OS by handling some portion of tasks' processing, service calls and task scheduling in hardware.
- The CheckSum accelerator automatically calculates "on-the fly" the 4-byte checksum placed at the end of Ethernet frames.
- With the Header EnDec accelerator, the CPU has a fast and direct read and write access to all frame header information again without any latency. Just for comparison, this data preprocessing typically requires about 15% of the overall CPU performance for a pure software based TCP/IP stack.
- Finally, the Buffer management accelerator automatically controls the buffer allocation and release functions for the Ethernet processing in the hardware.

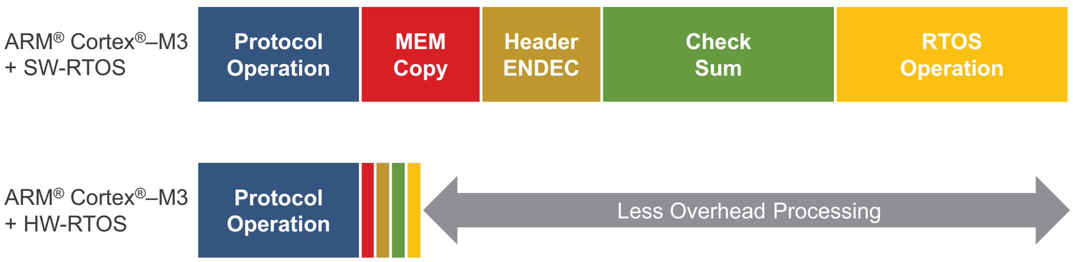

Figure 1: FTCP/IP Communication Overhead Processing

These accelerators allow the RTOS and Ethernet functions explained above to be directly executed in silicon without any involvement of the Cortex®-M3 CPU, thereby saving CPU performance and power dissipation during high Ethernet traffic operations. The power optimized Cortex®-M3 core supported by the variety of accelerators allows for a low power operation (~300 mW) of the communication engine. Furthermore, the accelerators improve the overall system timing with their deterministic and low-jitter processing speed. This is paramount for high-speed real time networks, as it enables better quality in an isochronous network environment.

Comparing an R-IN Engine with and without accelerators, an immense performance improvement can be observed. Whereas an R-IN Engine without accelerators operates at 100% CPU load at a UDP communication rate of 60 Mbps, an R-IN Engine with accelerators has only 30% CPU load at higher UDP communication rates of 95 Mbps. This leaves enough performance bandwidth for a simple IO application on the same core.

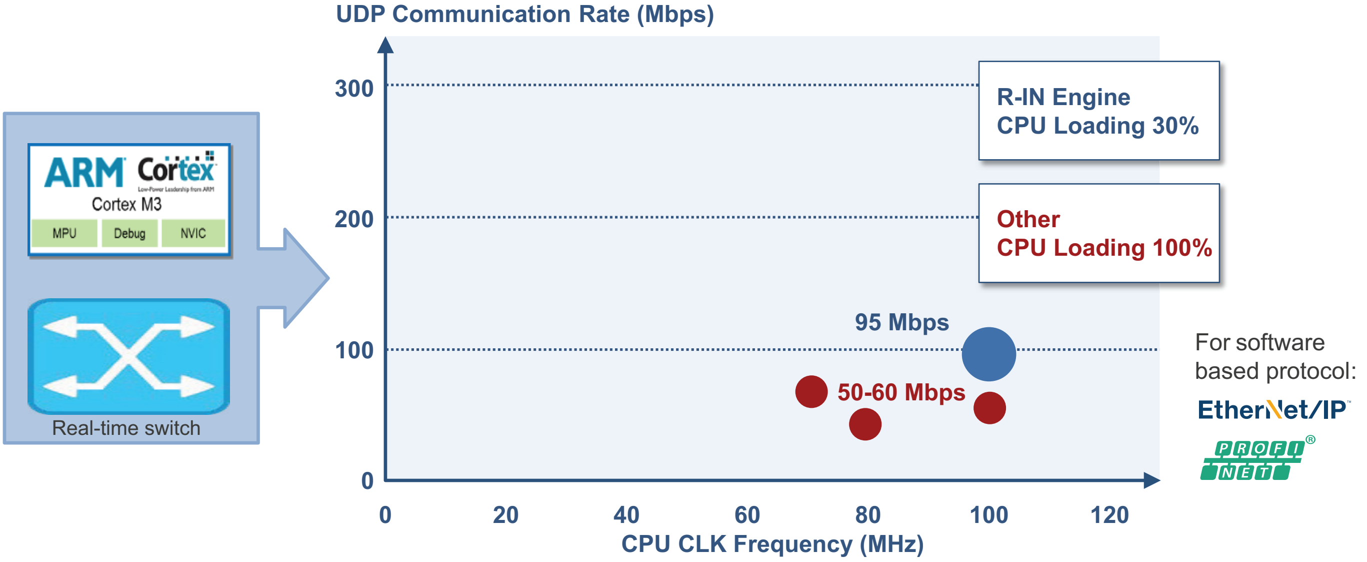

Figure 2: Real-Time Behavior with HW-Accelerators

Both the initial market success as well as the positive feedback from customers resulted in several R-IN Engine implementations. The first implementation was with the R-IN32M3. Three years later, in 2015, the motion control ASSP RZ/T1 was introduced to the market. Just as the R-IN32M3, the RZ/T1 supports EtherCAT, EtherNet/IP, Modbus TCP, PROFINET and TCP/IP communication protocols. Finally, the newly introduced RZ/N1 family features an upgraded version of the R-IN Engine, with 5 port real-time Ethernet switch and additional support for POWERLINK and Sercos industrial Ethernet protocols as well as the HSR/PRP redundancy protocols.

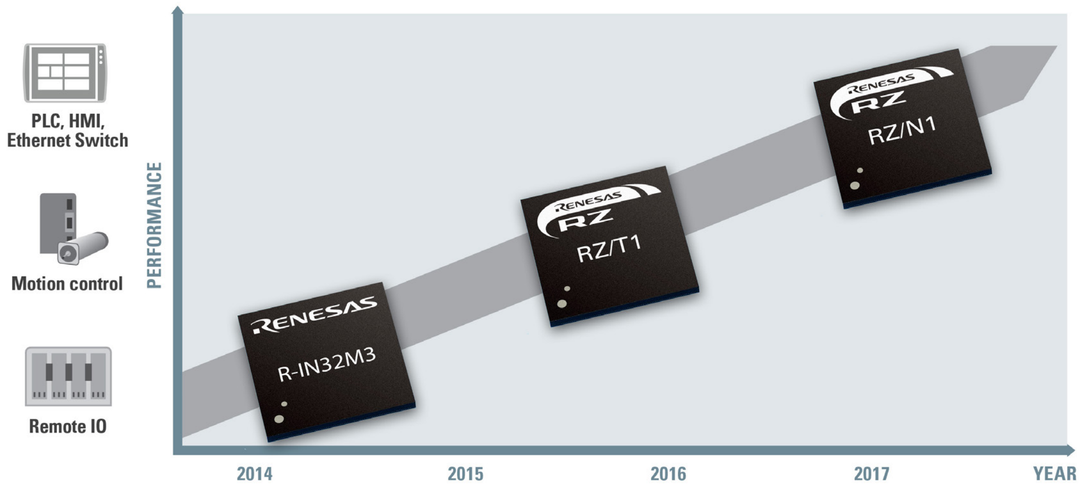

Figure 3: RZ/N1 – 3rd Generation R-IN Engine

R-IN Engine Family – Performance Meets Power Efficiency

With the introduction of the new RZ/N1 family, Renesas brings more performance to typical industrial communication applications. The RZ/N1 family is a System on Chip (SoC) with integrated application and communication block on a single chip. The integration of the application and communication block on a single chip is a general trend, which, due to the required deterministic behaviour makes even more sense in case of the RZ/N1 The closer the interaction between the application and communication blocks within a common hardware and software architecture, the easier it gets to achieve extreme real-time performance and determinism.

The key focus of RZ/N1 multi-protocol communication SoC is on the demanding applications in factory automation. With its' multi-core architecture, the RZ/N1 family offers enough performance for controllers, HMI solutions or industrial switches with industrial Ethernet support. Following the multiprotocol communication SoC R-IN32M3 for slave applications and the 3-in-1 SoC RZ/T1 for motion control applications, the RZ/N1 is a logical extension of the Renesas industrial automation portfolio and gives a scalable and software compatible platform for industrial automation applications. Furthermore, thanks to its sophisticated embedded 5-port switch, RZ/N1 brings cutting-edge performance in industrial networking, enabling features required for a TSN end-point like traffic shaping, time synchronization and frame classification/prioritization.

The RZ/N1 family consists of 3 products viz. the RZ/N1D, the RZ/N1S and the RZ/N1L. Both RZ/N1D and RZ/N1S have one application block and one communication block integrated on the same die. For a seamless data exchange, these blocks work hand-in-hand, connected via a communication API (Application Programming Interface) based on the GOAL abstraction layer. Both the core-2-core communication and the R-IN Engine are transparent for the application developer allowing simpler application development.

The application block of the RZ/N1D SoC features a dual ARM® Cortex®-A7 and is suitable for high-end applications such as network switches, PLCs, and gateways. The product is available in two packages viz. the 400 BGA and the 324 BGA. The application block of the RZ/N1S SoC features a single ARM® Cortex®-A7 and is intended for mid-range applications such as nano-PLCs and operator terminals. RZ/N1S is also available in two packages viz. the 324 BGA and the 196 BGA. The communication block of both RZ/N1D and RZ/N1S is nearly identical and depending on the product option, offers 3 Ethernet ports in the smaller package or 5 Ethernet ports in the larger package.

The RZ/N1L has no dedicated application block, as it only features the Cortex®-M3 CPU. The idea is to offer a simple communication device for customers who want to bring network functionality to legacy devices. The RZ/N1L is available in 196 BGA package.

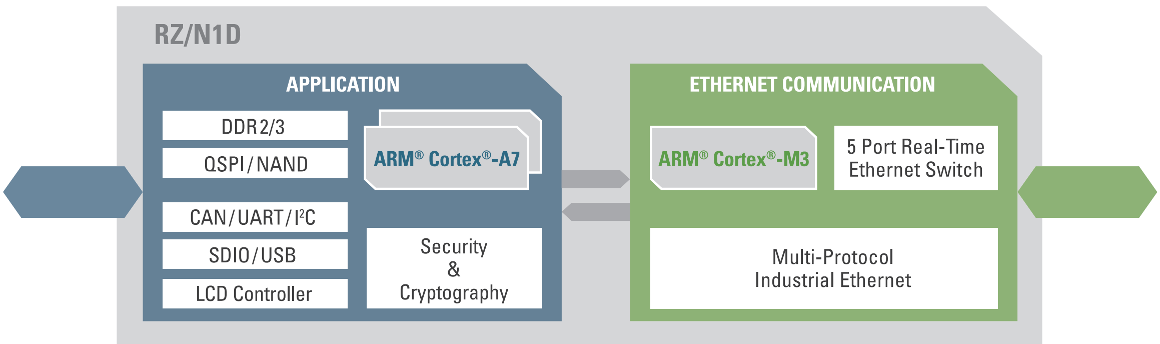

Figure 4: RZ/N1D Applications and Communication

Innovative Software – All Protocols under one Umbrella

Only a few years ago, it was sufficient for a device in an automation network to measure the data and forward it to the controller via Fieldbus protocol. This was usually handled in the hardware. As the industrial Ethernet technology started taking over the communication on the factory floor, the embedded firmware got more complex. New protocols such as EtherNet/IP, PROFINET, and several others are now handled in software. Moreover, new hardware components such as the embedded Ethernet switches need to be supported by the software. This increased system complexity exceeds the time-to-market requirement with ease, a key critical factor for success.

There is a growing pressure on the organizations to develop more functionality in less time, so the software has to be reused across different hardware platforms as a measure to save time, money and resources. And a key prerequisite for easy reuse of code is its modularity, high cohesion and coherence.

One of the key issues in this constellation is the variety of industrial Ethernet technologies like EtherCAT, EtherNet/IP, PROFINET and many others that are available on the market today. All these technologies have a long history, and there are functional differences between them. Some of them even require dedicated hardware support. This is why it is quite challenging to achieve the high cohesion and thus, it is no surprise if for example an application written for the PROFINET, needs to be completely rewritten for an EtherCAT implementation. Furthermore, with some hardware platforms available on the market, different protocols run on completely different ecosystems, which make the change between protocols very time, money and resource consuming.

One solution to this problem comes in the form of GOAL middleware in the RZ/N1 family. The idea behind it is to provide a software framework, which enables the develop-ment of low coupled software design. The GOAL API provides the interface to all components of the system and it is stable across all supported platforms regardless of the underlying hardware. The interface to the underlying system itself is done by the GOAL Target API which is fully transparent to the programmer.

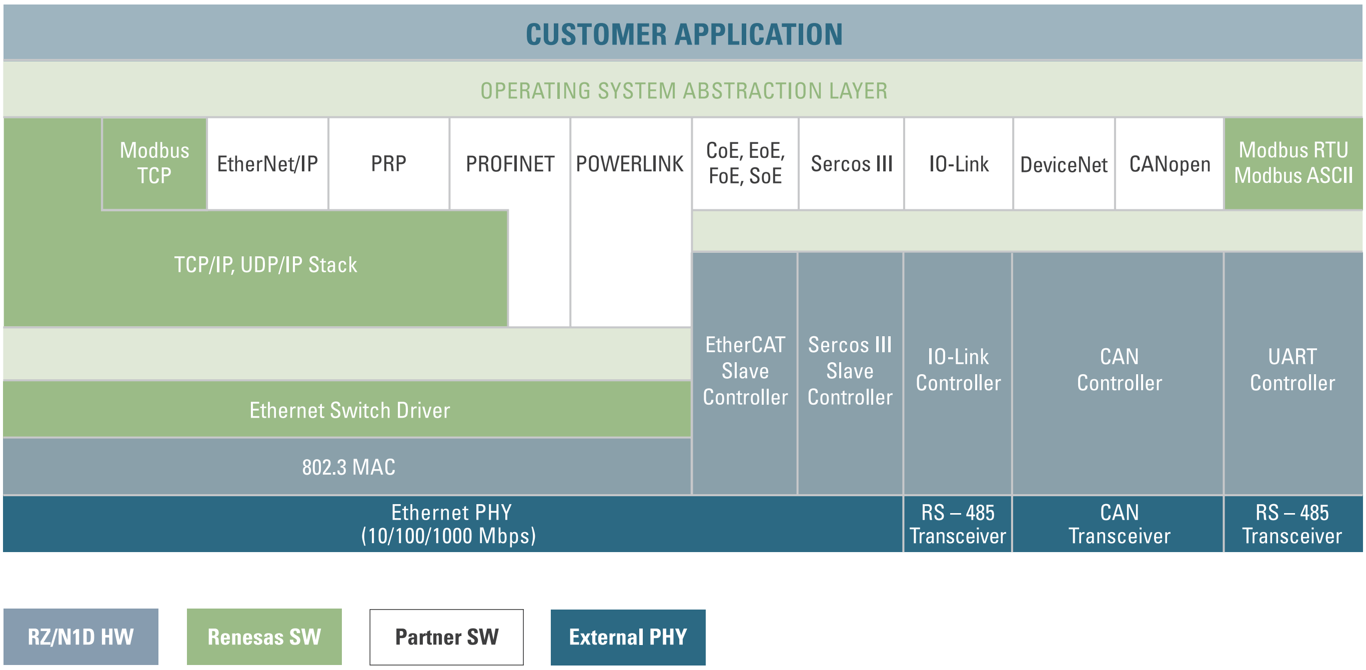

Figure 5: OSAL/GOAL – Easy Migration between Protocols and SoC

One of the key benefits of the GOAL platform is the structure of the Application Programming Interface (API) as a unique software communication interface for numerous industrial Ethernet protocols. All the protocols that are ported over GOAL, share the same abstraction layer and are connected with the application software over a unified software interface. Of course, the application software has a certain dependence on the stacks and has to be amended for each different communication protocol. As explained above, the protocol stacks contain similar but not same functions and parameters. However, GOAL allows an easy exchange of the communication technologies with minimal impact to the application software. An industrial networking application implemented under the GOAL framework has only to be modified minimally when porting it from one industrial protocol to another, thus giving the developer the advantage to focus his/her efforts in other development/testing areas than porting.

Another benefit of the GOAL platform comes from GOAL's inter-processor communication API. It allows connectivity between a communication processor (like RZ/N1L or R-IN32M3) with a legacy API. In the case of the highly integrated RZ/N1D and RZ/N1S, this API takes care of the communication between the communication and the application block. The core-2-core API is designed in a way that is independent of the underlying hardware system that transmits the messages. Shared memory, mailbox based systems or SPI – the communication can be adapted to any of these. As a result, applications written on top of the GOAL API can be used across all these different hardware components! Since the GOAL target interface is provided for all R-IN engine based platforms, developers can use the same software on multiple R-IN Engine based SoCs, R-IN32M3, RZ/T1 or the RZ/N1 family.

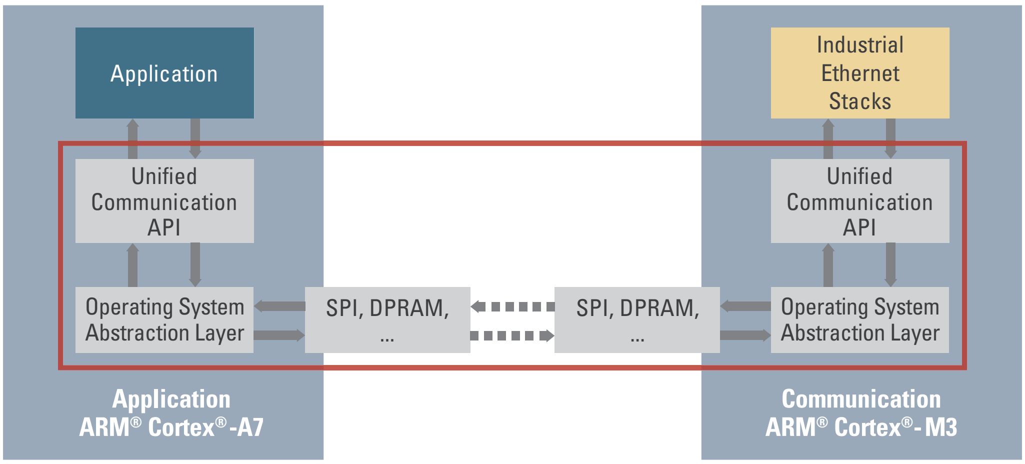

Figure 6: GOAL-based Core2Core Communication

While companies with dedicated industrial communication resources have the flexibility to even port own protocol stacks over GOAL, companies with limited resources can simply regard the R-IN Engine within the RZ/N1 as an industrial Ethernet black-box. Together with our business partners like Cannon Automata, Net Module, port GmbH, and TMG TE who are market leaders in the field of industrial communication software, we have already ported numerous protocol stacks to the R-IN Engine. The GOAL API is available directly on the Cortex®-A7 side, where the application software resides.

Besides the Industrial Communication part, the GOAL is equipped with a TCP/IP stack independent web server and comes with supporting network protocols such as SNMP and RSTP. This makes GOAL the ideal basis for network aware devices such as managed switches, routers and gateways. On top of the TCP/IP stack GOAL provides an independent stable API for TCP/IP based applications with support for a variety of TCP/IP stacks regardless whether the stacks support the BSD socket interface or not.

The rest of the RZ/N1 ecosystem depends on the product version. The RZ/N1D application block runs under the Linux OS or ThreadX. The RZ/N1S, on the other hand supports ThreadX and VxWorks. The R-IN Engine in the communication block of all three devices runs with ultron libraries.

A solution kit based on RZ/N1D and GOAL is already available and can be ordered directly at the Renesas webshop or at one of the distribution partners. For more information please visit the RZ/N1 web page: www.renesas.com/rzn

Written by: Ognen Basarovski, Renesas Electronics (Europe) GmbH

Document No. R01PF0148ED0100