Linux Kernel Support for IEEE 1588 Hardware Timestamping

Michael Rupert, Principle System Design Engineer, Data Center Business Division

Abstract

This white paper discusses Linux kernel support for IEEE 1588 hardware timestamping, including the precision time protocol (PTP) hardware clock (PHC) infrastructure, and the SO_TIMESTAMPING socket option. The PHC infrastructure and the SO_TIMESTAMPING socket option offer standardized APIs for user-space applications and Linux kernel device drivers. These kernel facilities improve device driver availability and simplify system integration.

Introduction

IEEE 1588 defines a protocol, the Precision Time Protocol (PTP), that enables accurate synchronization over packet switched networks (PSN). Hardware timestamping of PTP event messages is key to achieving nanosecond synchronization accuracy for PTP slave clocks.

A typical PTP slave clock with hardware time stamping can be divided into four components:

1. User-space software that implements an IEEE 1588 protocol stack and a PTP clock servo
2. Hardware time stamp unit (TSU) integrated with a MAC or PHY
3. PTP hardware clock (PHC) that provides the timing reference for the hardware TSU and that is controlled by the PTP clock servo
4. Linux kernel

The Linux kernel implements built-in support for hardware timestamping of PTP event messages. The support is comprised of the PHC infrastructure and the SO_TIMESTAMPING socket option. These kernel facilities provide standardized user-space APIs for the PHC and TSU functions, and they provide standardized interfaces for PHC and TSU device drivers. The net result for system integrators is improved availability of device drivers and simplified system integration, resulting in lower development costs and reduced time to market.

To realize the benefits of the PTP support in Linux, PHC device drivers must support the PHC infrastructure; and TSU drivers must support the SO_TIMESTAMPING socket option.

Renesas provides a Linux kernel driver for the ClockMatrix family that supports the Linux PHC infrastructure. Any IEEE 1588 software package that uses the Linux PHC API can control a ClockMatrix device using this API. For example, Linux PTP from the Network Time Foundation is designed to use the Linux PHC API. The ClockMatrix Linux kernel driver is available as part of main line Linux.

The Renesas PTP Clock Manager software includes a PTP clock servo with a packet delay variation (PDV) filter and other functionality that meets global ITU-T synchronization recommendations for telecom applications. The PTP Clock Manager can be used with any IEEE 1588 protocol stack (e.g., Linux PTP) and it is compatible with the Linux PHC API. The PTP Clock Manager is available under license from Renesas.

Contents

Definitions

- **Clock**
  IEEE 1588 defines a clock as follows: A device that can provide a measurement of the passage of time since a defined epoch.
- **Clock Signal**
  IEEE 1588 defines a clock signal as follows: A physical signal that has periodic events. The periodic events mark the significant instants at which a time counter is incremented. The clock signal is characterized by its frequency and phase.
- **IEEE 1588**
  Precision Time Protocol – IEEE 1588[1] defines a protocol, the Precision Time Protocol (PTP), that enables accurate synchronization over packet switched networks (PSN). PTP has a wide range of applications including telecom, industrial, enterprise, and others. PTP can be used to synchronize PTP slave clocks with nanosecond accuracy.

Basic Synchronization for a PTP Slave Clock

Figure 1 shows the basic PTP event message exchange between a PTP master and a PTP slave; also shown is a PTP slave clock implemented as a servo loop with a clock servo controller (clock servo). Under ideal conditions, the clock servo can calculate the time offset of its local time stamp unit (TSU) versus the TSU of the master using EQ1. The time offset is digitally compensated by the clock servo; afterwards, the frequency of the PTP slave clock signal is periodically adjusted to keep the time offset near zero. Therefore, the PTP slave clock is synchronized and locked to the PTP master clock.

EQ1. $$PTP\ Slave\ Clock\ Time\ Offset = \frac{(t_2 - t_1) - (t_4 - t_3)}{2}$$

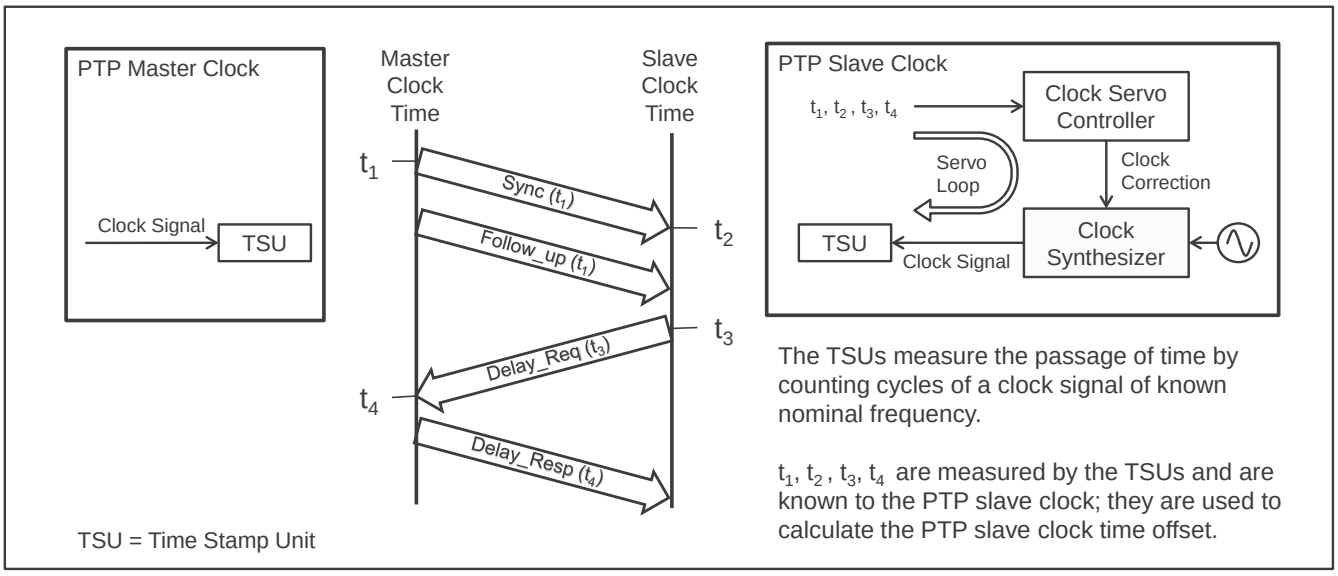

Figure 1. Basic Precision Time Protocol Event Message Exchange

[1] IEEE Std 1588-2019, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.

Hardware Timestamping of PTP Event Messages

Hardware timestamping of PTP event messages is key to achieving nanosecond synchronization accuracy for PTP slave clocks. The TSU should be in the signal path as close to the physical layer as practical. Hardware TSUs are typically implemented in network interface devices such as MACs or PHYs. Compared to software timestamping, hardware timestamping reduces uncertainty in the arrival and departure times of PTP event messages from milliseconds to nanoseconds; and in this way, hardware timestamping improves the accuracy of a PTP slave clock.

Hardware TSUs incorporate a time of day (TOD) accumulator that measures the passage of time by counting the cycles of a reference clock signal from a PTP hardware clock (PHC). The PHC is steered by the PTP clock servo that issues corrections (clock operations) to the PHC. The IEEE 1588 protocol stack obtains the PTP timestamp information from the hardware TSU and provides it to the clock servo. This is illustrated in Figure 2.

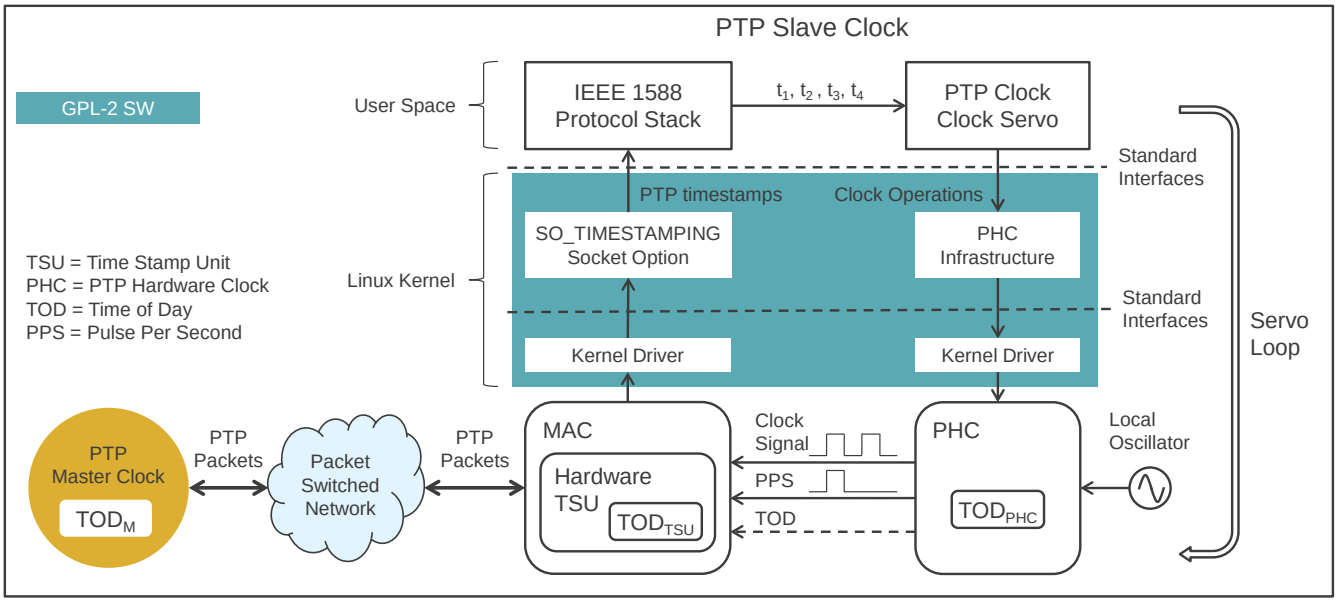

Figure 2. Typical PTP Slave Clock Implementation Using Hardware Timestamping

The logical PHC and TSU functions can be implemented in a single silicon device or they can be implemented in separate devices. Separate timing devices have advantages[2] as a central resource for generating and managing clock signals that are shared between TSUs in multiple devices. For this reason, the PHC function is often implemented in a dedicated timing device. Figure 2 shows TSU and PHC functions implemented in separate devices.

A perennial challenge for PTP system integrators is obtaining and maintaining drivers for PHCs and hardware TSUs. It is inefficient for silicon device manufacturers to create drivers to cover more than a few of the PTP protocol stacks and clock servo software packages available in the industry. For this reason, device manufacturers typically provide example driver code and leave driver development to the system integrator.

The Linux kernel implements built-in support for hardware timestamping of PTP event messages. The support is comprised of the PHC infrastructure and the SO_TIMESTAMPING socket option (see Figure 2). These kernel

[2] Advantages include: Adjusting clock skews with sub-nanosecond resolution, measuring clock delays, generating low phase-noise clocks, generating clocks with a range of frequencies, generating ITU-T compliant synchronous Ethernet clocks, and managing redundant clock sources, etc.

facilities provide standardized user-space APIs for the PHC and TSU functions, and they provide standardized interfaces for PHC and TSU device drivers.

Developers can write their user-space PTP applications for the standard Linux APIs without detailed knowledge of the PHC or TSU devices that will be used in the final system. Similarly, silicon device manufacturers can write and validate Linux kernel device drivers using the standard Linux kernel APIs without detailed knowledge of the user-space applications that will use them. The net result for system integrators is improved availability of device drivers and simplified system integration, resulting in lower development costs and reduced time to market.

Linux kernel support for PTP helps ensure forward compatibility with new kernel releases. The Linux kernel developers strive to maintain stable user-space APIs as the kernel evolves, and kernel device drivers that are part of main line Linux are maintained from release to release. Furthermore, the Linux community and the silicon device manufacturers can make improvements to the drivers and every system that uses them can easily adopt them.

To realize the benefits of the PTP support in Linux, PHC device drivers must support the PHC infrastructure; and TSU drivers must support the SO_TIMESTAMPING socket option. For reasons described below, it is also useful for the TSU driver to support the PHC infrastructure for its TOD accumulator.

The SO_TIMESTAMPING socket option and the PHC infrastructure are available together in Linux kernel versions 3.0 and later.

Synchronizing a TSU with a Separate PHC

When the PHC and TSU are implemented separately, the system integrator must synchronize TOD_{TSU} with TOD_{PHC}. This can be done by timing TOD_{TSU} with the same PHC clock and pulse per second (PPS) signals that time TOD_{PHC}. At system start-up there will be a static time offset between TOD_{TSU} and TOD_{PHC} that must be eliminated; there is an open-source user-space application called ts2phc[3] available for this purpose. The ts2phc utility controls TOD_{TSU} as a PHC; a prerequisite is that the TSU driver must support the PHC infrastructure. Any frequency adjustments made to the PHC after the two TODs are aligned will be tracked by the TSU.

Renesas PTP Synchronization Solutions

The Renesas ClockMatrix family of timing devices includes several products designed for use as PHCs (e.g., 8A34001/2/3/4/5 are excellent devices for telecom PHC applications). Renesas created a Linux kernel driver for the ClockMatrix family that supports the Linux PHC infrastructure. The ClockMatrix driver is available as part of main line Linux and it is compatible with Linux kernel versions 3.0 and higher.

PTP Synchronization Solution Using Open Source Software Only

The ClockMatrix Linux kernel driver is compatible with the Linux PTP software suite version 3.0 and higher, including its IEEE 1588-2019 protocol stack, and the included PTP clock servo and ts2phc utility. Linux PTP is an open-source (GPL-2) software suite available from the Network Time Foundation.

When used with a ClockMatrix device, Linux PTP will bypass the software low-pass filter function of its PTP clock servo and instead will use the write phase mode and hardware low-pass filter implemented by the ClockMatrix devices. This servo implementation is suitable for applications that do not need to handle high levels of packet delay variation (PDV)[4].

[3] The ts2phc utility is included in the Linux PTP software suite available from the Network Time Foundation.

[4] For more information, see the Renesas whitepaper titled *Limiting IEEE 1588 Slave Clock Wander Caused by Packet Delay Variation*.

Figure 3 illustrates a PTP slave clock design that uses open-source software only, and is suitable for applications that do not need to handle high levels of PDV.

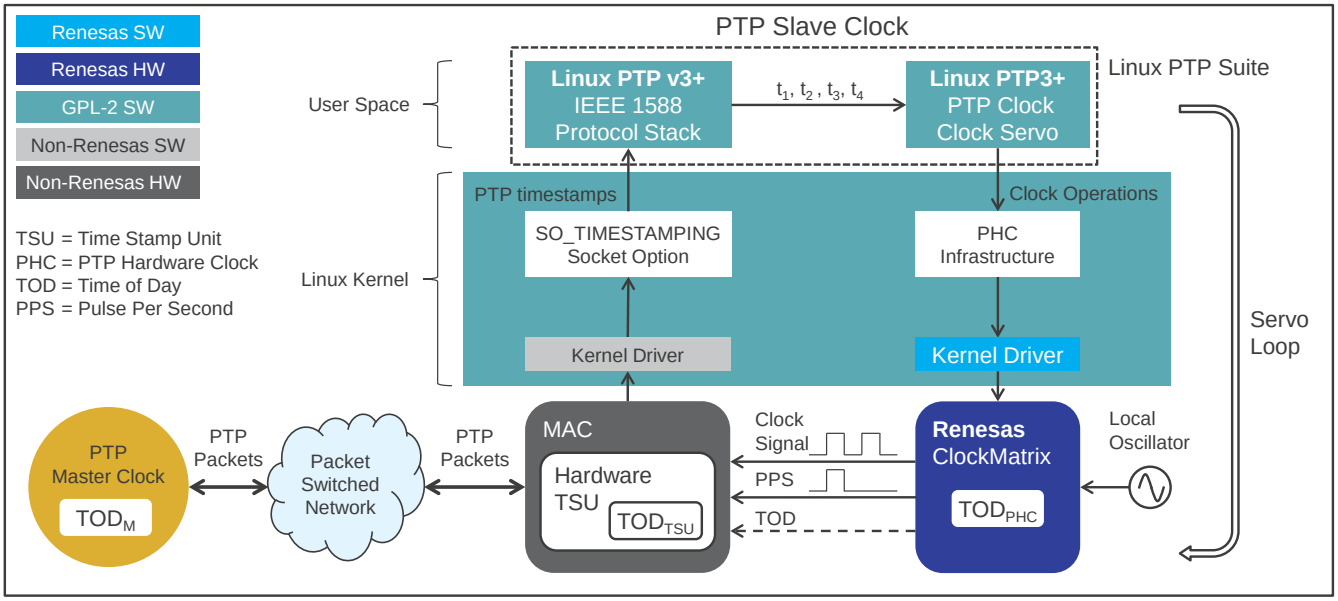

Figure 3. Renesas PTP Slave Clock Implementation Using Only Open-Source Software

PTP Synchronization Solution Using Open Source and Renesas Software

The design shown in Figure 3 is suitable for many PTP applications. However, for applications that require tolerance to PDV and global compliance to ITU-T synchronization recommendations for telecom applications, a more sophisticated PTP clock servo is needed.

The Renesas PTP Clock Manager user-space application is available from Renesas under license for use with Renesas timing devices. The PTP Clock Manager includes a PTP clock servo with a PDV filter[4] and other functionality that meets global ITU-T synchronization recommendations for telecom applications. The PTP Clock Manager is compatible with Linux PTP versions 3.0 and higher, specifically the IEEE 1588-2019 protocol stack; it is also compatible with Linux kernel versions 3.0 and higher.

The Renesas PTP Clock Manager synchronizes the TOD_{TSU} with the TOD_{PHC} using its own alternative to ts2phc. While synchronizing TOD_{TSU} with TOD_{PHC}, the PTP Clock Manager controls TOD_{TSU} as a PHC; a prerequisite is that the TSU driver must support the PHC infrastructure.

Figure 4 shows a PTP slave clock design that uses the Renesas PTP Clock Manager software. This clock design is suitable for applications that need to manage high levels of PDV and/or that need to comply with global ITU-T synchronization recommendation for telecom applications.

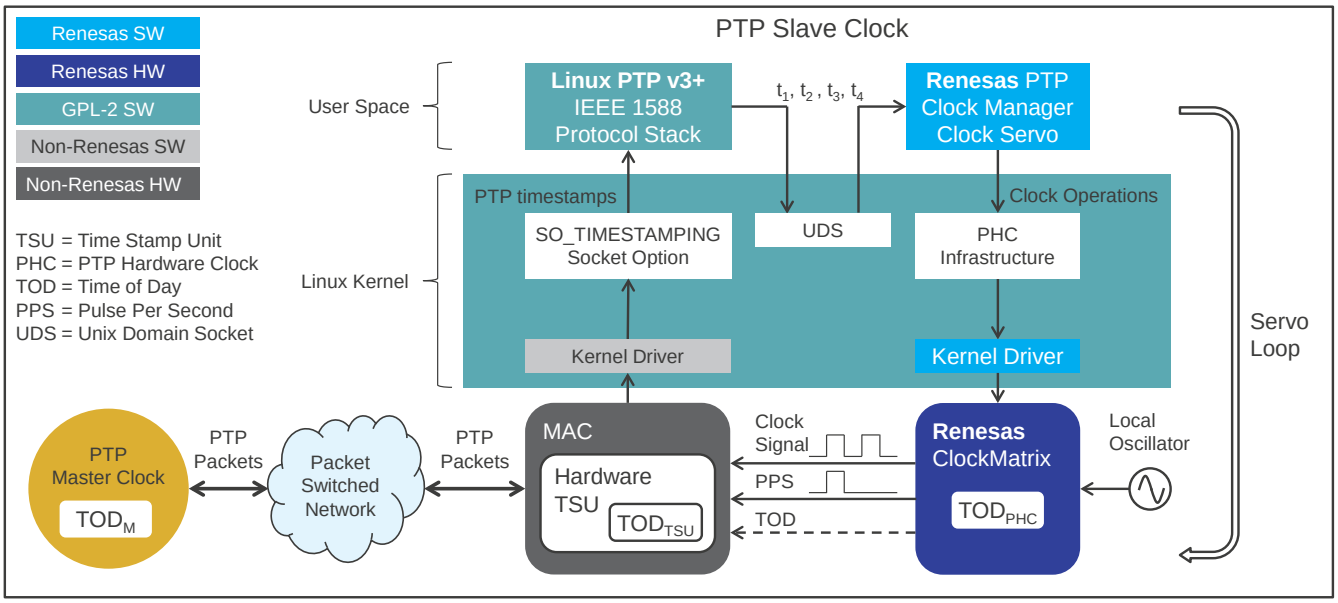

Figure 4. Renesas PTP Slave Clock Implementation Using PTP Clock Manager Software

Conclusion

Hardware timestamping of PTP event messages is key to achieving nanosecond synchronization accuracy for PTP slave clocks. Hardware timestamping is implemented with hardware TSUs timed by clock signals generated by PHCs. The hardware TSUs are often integrated with PHYs and MACs for proximity to the physical layer.

The logical PHC and TSU functions can be implemented in a single silicon device or they can be implemented in separate devices. A separate timing device has advantages as a central resource for generating and managing clock signals that are shared between TSUs in multiple devices. For this reason, the PHC function is often implemented in a dedicated timing device.

The Linux kernel implements built-in support for hardware timestamping of PTP event messages. The support is comprised of the PHC infrastructure and the SO_TIMESTAMPING socket option. These kernel facilities provide standardized user-space APIs for the PHC and TSU functions, and they provide standardized interfaces for PHC and TSU device drivers. The net result for system integrators is improved availability of device drivers and simplified system integration, resulting in lower development costs and reduced time to market.

To realize the benefits of the PTP support in Linux, PHC device drivers must support the PHC infrastructure; and TSU drivers must support the SO_TIMESTAMPING socket option. It is also useful for the TSU driver to support the PHC infrastructure for its TOD accumulator.

The Renesas ClockMatrix family of timing devices are an excellent choice to implement PHCs for hardware timestamping applications. Renesas provides a Linux kernel driver for the ClockMatrix family that supports the Linux PHC infrastructure.

The Renesas PTP Clock Manager software implements a PTP clock servo with a PDV filter and other functionality that meets global ITU-T synchronization recommendations for telecom applications. The PTP Clock Manager is compatible with the Linux PHC infrastructure.

The standardized PHC infrastructure and SO_TIMESTAMPING socket option APIs enable modularity of PTP functions. For example, the Renesas PTP Clock Manager can be substituted for the Linux PTP servo for applications that require PDV filtering. Similarly, the hardware based low-pass filter in the ClockMatrix devices can be substituted for the software based low-pass filter in the Linux PTP servo.

Revision History

Revision	Date	Description
1.0	Jan.29.21	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.