RISC-V Days Japan Tokyo 2022 Spring Demonstration

# A RISC-V Wireless IoT running Amazon FreeRTOS Integrates Marmot System with digital caliper and Over-the-Air Software Update

May 31, 2022

SH Consulting Group (SHC)

## Overview

This demonstration shows a full integration of secure OTA firmware upgrade and digital caliper monitor for an IoT with a 32-bit RISC-V running FreeRTOS on Marmot System. In this demonstration the RISC-V RTOS-based IoT conducts OTA via LTE Mobile SIM leveraging Amazon Web Services (AWS) IoT Core.

When IoT devices are deployed, the IoT service providers need to check the identity of the IoT device that is communicating with the messaging gateway. When a device is powered up for the first time, the hardware root of trust (RoT) generates key pairs for the devices, which are then used to authenticate and encrypt the traffic. The keys are generated inside the hardware root of trust (RoT) itself and are thereby protected from being retrieved by external programs.

SHC demo, after generating key pairs, utilizes the FreeRTOS MQTT library and OTA agent to connect to the AWS Cloud and then performs OTA firmware update function to update new firmware to RISC-V SoC, RISC-V Microcontroller and other microcontroller.

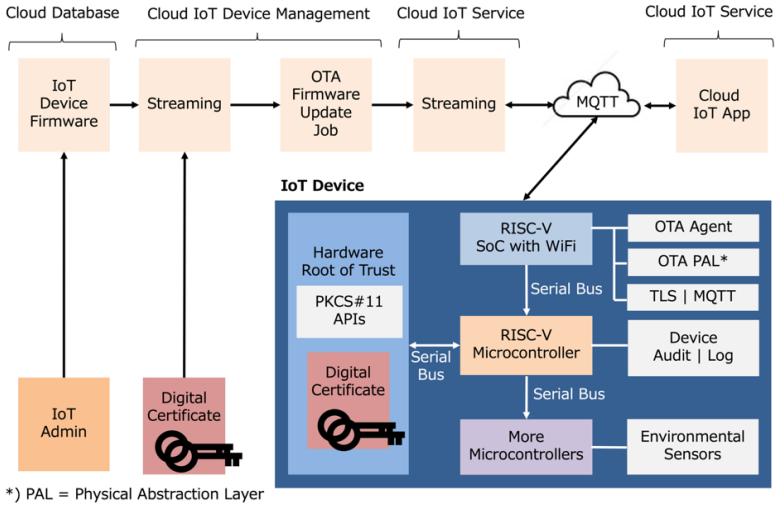



**Figure 1: Full Diagram of a RISC-V AWS IoT Core Device Example**

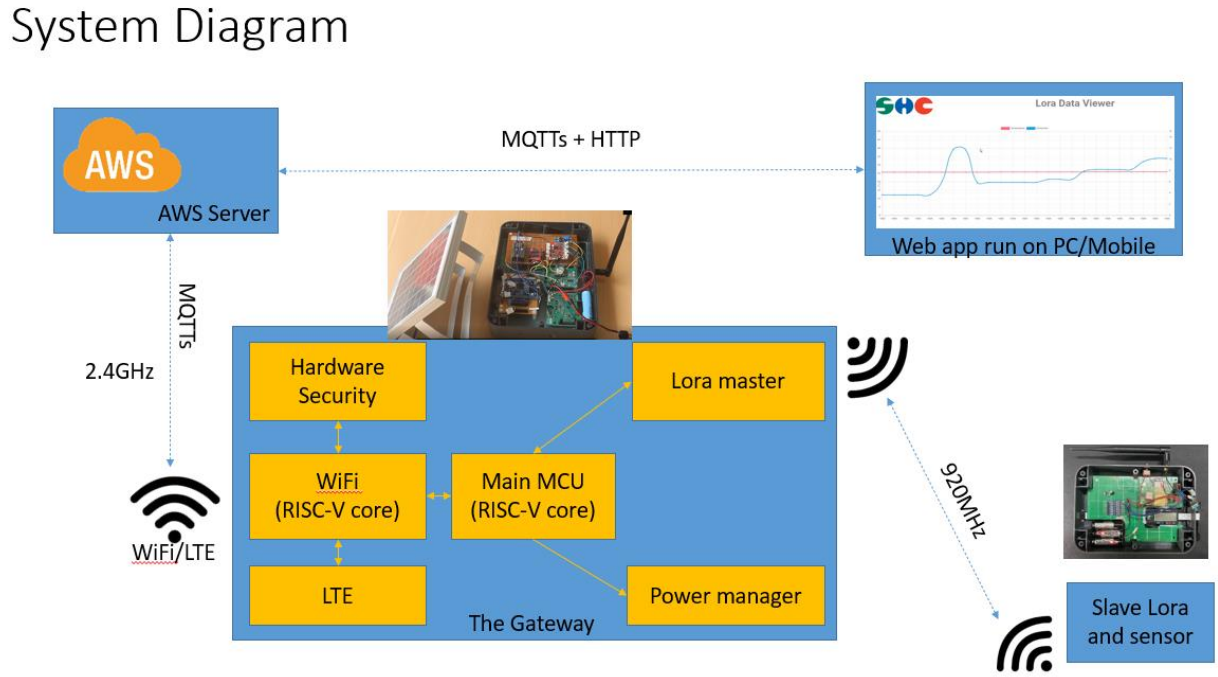
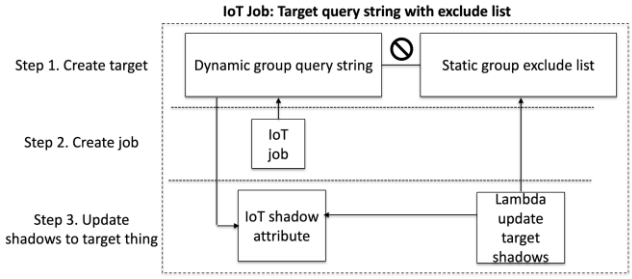



**Figure 2: RISC-V OTA and RoT Demonstration Setup**

Our RISC-V FreeRTOS OTA demo is continued after the first demo in Tokyo day recently in Autumn 2021. It is one of a series of demonstrations for microcontroller system in which OTA for multiple processors is performed in a secure manner. In previous demos we shown OTA for 1st and 2nd RISC-V SoC. This time we are continuing with OTA demo for 3rd Microcontroller.

## Over-the-Air(OTA) Firmware Update

Over-the-air (OTA) firmware update allows a user to remotely update firmware to a device and redeploy the device with new firmware automatically without physical intervention. A new set of firmware is digitally signed so the system can verify the signature when it receives the file. When a programmer creates an OTA update, the OTA Update Manager service creates an AWS IoT job to notify the device that an update is available. The OTA demo application runs on the device with a FreeRTOS task that subscribes to notification topics for AWS IoT jobs and listens for update messages. When an update is available, the OTA Agent publishes requests to AWS IoT Core and receives updates using the HTTP or MQTT protocol. In SHC demo MQTT protocol is used. The OTA Agent checks the digital signature of the downloaded files, if the files are valid, installs the firmware update. After finishing firmware updating, the board is automatically reset to start new firmware.




```
Partition Table:
## Label            Usage          Type ST Offset   Length
0 nvs              WiFi data        01 02 00010000 00006000
1 otadata          OTA data         01 00 00016000 00002000
2 phy_init         RF data          01 01 00018000 00001000
3 ota_0            OTA app          00 10 00020000 00177000
4 ota_1            OTA app          00 11 001a0000 00177000
5 storage           WiFi data        01 02 00317000 00010000
End of partition table
```

**Figure 3: AWS OTA Firmware Update Process (Left) and OTA Partition Table (Right)**

RISC-V Days Japan Tokyo 2022 Spring Demonstration

## Measurements Made from RISC-V OTA Firmware Update

**RISC-V Wifi Module**: 160MHz RISC-V running QSPI flash at 80MHz update 1,151,488 bytes of ESP32C3 firmware image in 281 blocks via MQTT took 4.416 min or 254 sec

**RISC-V Flash MCU**: 8MHz RISC-V core running in internal flash update 119,108 bytes of GD32VF103 firmware image in 466 block (256 byte/block) via MQTT and UART took 17.3 min or 1038 sec

**Lora Master MCU**: 47.972352 MHz ARM cortex M0+ core running in internal flash update 47,744 bytes of MKL16 firmware image in 48 sectors (1 Kbyte/sector) via MQTT and UART took 7.766667 min or 466 sec

## Memory Requirements

The system connect with AWS IoT Core using a commercial hardware root of trust (RoT). It is required memory larger than 219.5Kb and random access memory (RAM) larger than 32.87Kb. In this demo the AWS IoT core connection is implemented in ESP32C3

**RISC-V Wifi Module:** To implement an OTA firmware update solution with confidence more than 1MB of flash memory or more is required. And RAM is required 400Kb

**RISC-V Flash MCU**: the memory is required 128Kb (10Kb for bootloader + 512 bytes for version store + 117.5 Kb for application) and 1Mb external memory (spi flash) to store the new firmware when esp32c3 send it via UART. The RAM is required 32Kb.

**Lora Master MCU**: the memory is required 128Kb (10Kb for bootloader + 256 bytes for version store + 54 Kb for application + 65279 bytes to store the new firmware when GD32VF103 send it via UART). The RAM is required 16Kb.

## **Marmot = M**icrocontroller **A**rchitecture to **R**esist **M**alware, **O**bstructions, and **T**ampering

Marmot Wireless IoT system is being developed. This is an RTOS IoT system which provides: (1) Fast Response Time, (2) Low Maintenance, (3) High Reliability, (4) High Security. Figure 4 shows a Marmot Emblem. Marmots are relatively large ground squirrels in the genus Marmota, with 15 species living in Asia, Europe, and North America. The handmade emblem was created to mark the boxes. Figure 5 shows Marmot systems.

**Figure 4: Marmot Emblem. Marmots are relatively large ground squirrels in the genus Marmota, with 15 species living in Asia, Europe, and North America. A handmade emblem was created to mark the boxes but this emblem idea was discarded because a solar panel covers Marmot enclosure and there is no proper visible place for an emblem.**

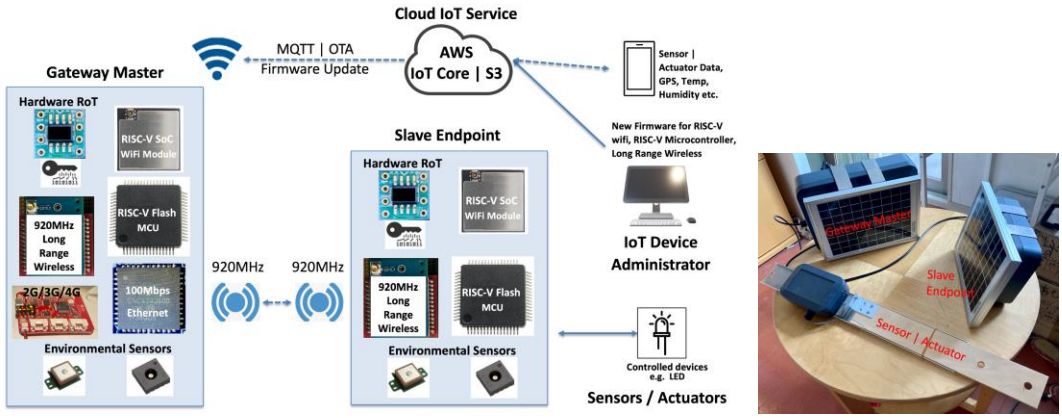



**Figure 5: Marmot Wireless IoT System Example**

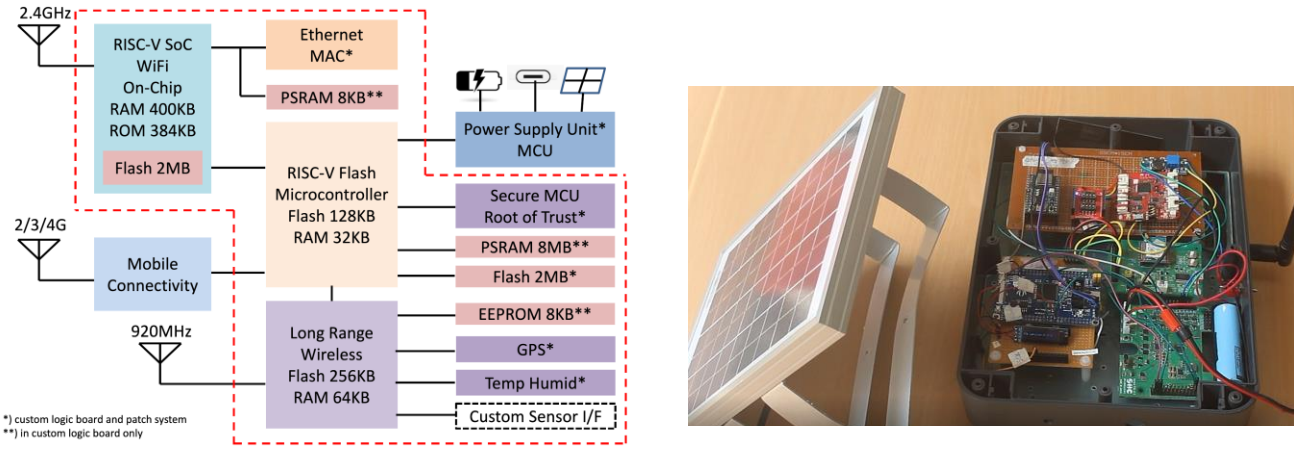



**Figure 6: Marmot System Diagram and Patch Boards used for Software Development**

## Cloud Service, Amazon AWS IoT Core

AWS IoT Core is a managed cloud service that lets connected devices interact with cloud applications and other devices. AWS IoT Core supports devices and messages, and processes and routes those messages to AWS endpoints and to other devices. With AWS IoT Core, applications can keep track of and communicate with all your devices, all the time, for OTA or pushing messages.

## RISC-V SoC Chip with WiFi / Bluetooth Connectivity

**RISC-V SoC** features Wi-Fi and Bluetooth Phy and is one of the RISC-V platforms qualified for Amazon FreeRTOS. The SoC comes with a 32-bit RISC-V Single-core processor running at 160MHz, 4MB external Flash, 384KB ROM and 400KB data SRAM, and GPIO, I2C, PWM, SPI, and UART. It has the following wireless features: (1) IEEE 11 b/g/n-compliant, (2) 1T1R mode with data rate up to 150Mbps, (3) MPDU and A-MSDU aggregation, (4) 4μs guard interval, (4) 2,412MHz to 2,484MHz operating frequency

## RISC-V Flash MCU

**RISC-V Flash MCU** is RISC-V platforms which 32-bit general-purpose microcontroller based on the RISC-V core, can run at up to 108MHz. it provides128 KB on-chip Flash memory and 32 KB SRAM memory and peripheral includes: 5x U(S)ART, 2 x I2C, 3 x SPI, 2 x I2S, 2 x CAN2.0, 1 x USBFS.

## Lora Master MCU

**Lora Master MCU** is MKL16Z128 which 32-bit general-purpose microcontroller based on the ARM cortex M0+ core, can run at up to 48MHz. it provides 128 KB on-chip Flash memory and 16 KB SRAM memory and peripheral includes: 2x UART, 2 x I2C, 2 x SPI, 1 x I2S, 1 x low power UART

## Hardware Root of Trust (RoT) Chip

A hardware root of trust (RoT) or a hardware root-of-trust (RoT) provides an IoT endpoint device security with countermeasures against physical tampering and side-channel attacks. The hardware root of trust (RoT) protects the private keys used by the IoT endpoint to establish the device identity, and prevents the private data from being taken out of the devices for impersonation and other malicious activities.

The particular hardware root of trust (RoT) device we used is designed to securely store multiple private keys along with their associated public keys and certificates. It supports random private key generation internally within the device to ensure that the private key can never be known outside of the device. The public key corresponding to a stored private key is always returned when the key is generated and may optionally be computed at a later time. The internal random number generator is designed to meet the requirements documented in the NIST 800-90A, 800-90B and 800-90C documents. These random numbers can be employed for usage as part of the device's crypto protocols.

Because each random number is guaranteed to be essentially unique from all numbers ever generated on this or any other device, their inclusion in the cryptographic protocol calculation ensures that replay attacks (i.e. re-transmitting a previously successful transaction) will always fail.

## Acknowledgements

This work is based on results obtained from project, JPNP16007, commissioned by The New Energy and Industrial Technology Development Organization (NEDO). Patents and trademarks pending for Marmot System.

## About SH Consulting Group

SH Consulting Group (SHC) has engineers in Vietnam and in Japan specialized in providing stability to RTOS, device drivers, and wireless connectivity for MCUs such as H8s, SHs, ARMs and RISC-Vs. It has been integrating OSes such as QNX, .NETMF, Linux, and Windows for MCUs and wireless solutions such as Lora, WiFi and Bluetooth for many years. They worked on Windows, Android and iOS platforms. In recent years SHC engineers enabled FreeRTOS for large semiconductor companies on ARM and RISC-V.

**SH CONSULTING K.K. (JAPAN)**
Tokyo Head Office: 7-18-13-502 Ginza, Chuo-ku, Tokyo, Japan 104-0061
Phone: 03-3833-3717
Tokyo Design Center: Room 202 Sunmail, 2-2038-13 Imokubo, Higashiyamato-shi, Tokyo 203-0033

**SH CONSULTING VIETNAM COMPANY LTD. (VIETNAM)**
(Local Name: CÔNG TY TNHH SH CONSULTING VIỆT NAM)
Quang Trung Software Park, Tan Chanh Hiep Ward, District 12 Ho Chi Minh City
Phone: 84-8-3715-0060