RX Family

Failure Detection and Movement Analysis Demonstration Using AWS Cloud and FFT

Introduction

This application note describes a demo system showing communication between an RX MCU and the Amazon Web Services (AWS) cloud service. The RX MCU is used to convert analog signals from a three-axis accelerometer into digital signals and perform digital signal processing (DSP) on them. The results are then sent to AWS, allowing for data analysis from remote locations.

For example, patient monitoring in the medical field or failure detection on manufacturing lines in the industrial field involve analysis of factors such as attitude and vibration. The capabilities of the demo system can be applied in fields such as these.

The sample program described in this application note consists of the following components:

- Inputting data by means of A/D conversion
- DSP (frequency analysis by fast Fourier transform (FFT) processing)
- Judgement of frequency analysis results
- Display on an LCD panel (frequency analysis results and judgement results)
- Sending of analysis results to AWS

The sample program described in this application note was designed using FreeRTOS. FreeRTOS services are used for things such as task switching and sending data to AWS.

Descriptions of the demo environment and procedure, and the sample program, are provided in the pages that follow.

![System Outline Diagram]

Visit the following webpage for information on boards, related programs, and development environments necessary for developing RX cloud connectivity solutions.

https://www.renesas.com/rx-cloud

This demo system requires additional parts and board rework. For details, refer to 3.2.2 Hardware Rework.

Target Device

RX65N Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.
Contents

1. System Overview......................................................................................................................... 4
  1.1 DSP (Frequency Analysis by FFT Processing) and Judgement of Frequency Analysis Results .... 5
  1.2 Display on LCD Panel ............................................................................................................. 6
  1.3 Sending Analysis Results to AWS.......................................................................................... 6
  1.4 Connecting to AWS .............................................................................................................. 7
  1.5 Operating Environment ......................................................................................................... 7

2. File Structure Associated with This Application Note ................................................................. 8
  2.1 File Structure Associated with This Application Note.......................................................... 8
  2.2 Structure of Sample Program .............................................................................................10

3. Running the Sample Program ......................................................................................................12
  3.1 Importing the Project .............................................................................................................12
  3.2 Connecting Equipment .........................................................................................................14
  3.2.1 Hardware Configuration ....................................................................................................15
  3.2.2 Hardware Rework ............................................................................................................16
  3.3 Network Environment and Certificate Issuance ...................................................................17
  3.4 Elasticsearch Service Domain Creation .................................................................................23
  3.5 Running the Sample Program and Confirming Operation ....................................................33
    3.5.1 LCD Display Details .......................................................................................................33
    3.5.2 Viewing Output Log in Tera Term ..................................................................................34
    3.5.3 Displaying Data Using the Elasticsearch Service ..........................................................36
  3.6 User-Modifiable Settings .......................................................................................................41

4. Description of Sample Program ..................................................................................................42
  4.1 Overview of Sample Program ...............................................................................................42
  4.2 Processing Sequence .............................................................................................................43
  4.3 Processing Flowchart ............................................................................................................45
  4.4 Details....................................................................................................................................46
    4.4.1 Normalization Processing..................................................................................................46
    4.4.2 FFT Processing ................................................................................................................47
    4.4.3 Judgement of FFT Processing Results .............................................................................48
    4.4.4 Data Transmission to AWS ............................................................................................48
    4.4.5 Viewing Communication Log Output in Tera Term .........................................................48
    4.4.6 Processing to Update Display on LCD Module ...............................................................49
    4.4.7 QoS Setting .....................................................................................................................49
  4.5 File Structure ........................................................................................................................50

5. Important Note ............................................................................................................................53
  5.1 Frequency Value of FFT Processing Result ............................................................................53
  5.2 Warning Message at Project Building ....................................................................................53
5.3 A Fee for the Elasticsearch Service ................................................................. 54

6. Reference ............................................................................................................. 55
   6.1 System Operation Monitoring in e² studio ......................................................... 55
   6.2 Memory Usage ................................................................................................. 58

7. Obtaining the Development Environment ......................................................... 59
   7.1 Obtaining e² studio ......................................................................................... 59
   7.2 Obtaining the Compiler Package ...................................................................... 59

8. Supplement ........................................................................................................... 59
   8.1 Notes on Using the Free Evaluation Version of the RX Family C/C++ Compiler Package ......................................................... 59
   8.2 DSP Library for RX Family ............................................................................. 59

9. Reference Documents .......................................................................................... 59

Revision History ...................................................................................................... 60
1. System Overview

Figure 1.1 shows an overview of the system described in this application note. DSP and sending of data to AWS are divided into separate tasks. Figure 1.2 shows the transitions between tasks.

This system uses a single RX65N MCU for all processing, from sampling of the input signals to judgement result output control.

The system principally performs control by using the following two tasks.

- **Task 1**
  Task 1 handles DSP, judgement of frequency analysis results, and display on the LCD panel. DSP processing proceeds in units of 1,024 samples. After one unit of processing completes, the peak frequency and peak frequency magnitude values are stored in the queue for transmission to AWS, and processing transitions to Task 2.

- **Task 2**
  Task 2 sends the peak frequency and peak frequency magnitude values to AWS. When data is stored in the queue the task transitions from the block state to the execution state and transmits the stored data to AWS. When transmission of data to AWS finishes, the queue is emptied and the task transitions back to the block state, and processing transitions to Task 1.
1.1 DSP (Frequency Analysis by FFT Processing) and Judgement of Frequency Analysis Results

When the DSP receives a sine wave as an input signal, it outputs a frequency spectrum like that shown in Figure 1.3. DSP performs normalization followed by FFT processing. For example, failure detection can be accomplished by connecting an accelerometer to the A/D converters, as shown in Figure 1.4. For instructions for connecting the accelerometer, refer to 3.2, Connecting Equipment.

The processing steps that take place from analog signal input through judgement are described below.

1. A/D Conversion

   A/D conversion is implemented using the 12-bit A/D converter module (S12AD), compare match timer (CMT), and event link controller (ELC) at a sampling frequency of approximately 1 kHz. First, the CMT generates compare match events in regular cycles of approximately 1 ms, and these events are used by the ELC as A/D conversion start triggers for the S12AD. The converted data is transferred to an input buffer by the DMA controller (DMAC). There are three input signals, and they are processed by three A/D converters.

2. Normalization

   The input signals converted by the S12AD are stored in the input buffer as 12-bit (unsigned) data. The 12-bit data stored in the input buffer is normalized (by bias processing and scaling) to 31-bit (signed) format.

3. FFT Processing

   The normalized data undergoes 1,024-point FFT processing, and the result is stored in an output buffer.

4. Judgement

   A pass/fail (OK/NG) judgement is made based on the frequency analysis results.

![Figure 1.3 Input Signal and FFT Processing Result](image)

![Figure 1.4 Input Signal Analysis Procedure](image)
1.2 Display on LCD Panel

Figure 1.5 shows an illustration of output to the LCD module.

The LCD module displays the three-axis FFT analysis results (peak frequency and peak frequency magnitude), the judgement result based on a comparison with the expected values, and the AWS and Wi-Fi connection status. The judgement result (Judge) is indicated as pass (OK) or fail (NG), and the Wi-Fi and AWS status as CONNECTED or DISCONNECTED. For the method of judging, refer to 4.4.3, Judgement of FFT Processing Results.

1.3 Sending Analysis Results to AWS

The data peak frequency and peak frequency magnitude values obtained by DSP are sent to AWS. Figure 1.6 shows the sequence from reception of the transmitted data to its display by the Elasticsearch Service. Transmission to AWS occurs immediately after frequency analysis using DSP. Transmission to AWS takes place at intervals of approximately one second, which corresponds to the DSP frequency analysis cycle duration.

The AWS services used by the demo are IoT Core and Elasticsearch Service. For the setting procedures, refer to 3.3, Network Environment and Certificate Issuance, and 3.4, Elasticsearch Service Domain Creation.
1.4 Connecting to AWS

The procedure for connecting to AWS involves first establishing a Wi-Fi connection and then connecting to AWS. If the necessary settings are made in a header file before running the demo program, the demo program will perform the connection procedure automatically when the program is run.

DSP frequency analysis and display of results on the LCD module continue even if the Wi-Fi or AWS connection is lost while the program is running. In this case, the Wi-Fi and AWS status shown on the LCD module changes from CONNECTED to DISCONNECTED.

1.5 Operating Environment

The operation of the sample program described in this application note has been confirmed under the conditions listed in Table 1.1.

Table 1.1  Operation Confirmation Conditions

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU</td>
<td>R5F565NEDDFP (RX65N Group)</td>
</tr>
<tr>
<td>Operating frequency</td>
<td>HOCO clock: 20 MHz</td>
</tr>
<tr>
<td></td>
<td>PLL circuit output: 240 MHz (HOCO clock × 12)</td>
</tr>
<tr>
<td></td>
<td>System clock (ICLK): 120 MHz (PLL circuit output × 1/2)</td>
</tr>
<tr>
<td></td>
<td>FlashIF clock (FCLK): 60 MHz (PLL circuit output × 1/4)</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (PCLKA): 120 MHz (PLL circuit output × 1/2)</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (PCLKB): 60 MHz (PLL circuit output × 1/4)</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (PCLKC): 60 MHz (PLL circuit output × 1/4)</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (PCLKD): 60 MHz (PLL circuit output × 1/4)</td>
</tr>
<tr>
<td></td>
<td>External bus clock (BCLK): 120 MHz (PLL circuit output × 1/2)</td>
</tr>
<tr>
<td></td>
<td>USB clock (UCLK): 48 MHz (PLL circuit output × 1/5)</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>3.3 V</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Single-chip mode</td>
</tr>
<tr>
<td>Processor mode</td>
<td>Supervisor mode</td>
</tr>
<tr>
<td>Integrated development environment</td>
<td>Renesas Electronics e² studio 2021-01</td>
</tr>
<tr>
<td>C compiler</td>
<td>Renesas Electronics RX Compiler CC-RX V3.03.00</td>
</tr>
<tr>
<td>Compiler options</td>
<td>-lang = c99</td>
</tr>
<tr>
<td></td>
<td>-save_acc</td>
</tr>
<tr>
<td>Endian order</td>
<td>Data: Little endian</td>
</tr>
<tr>
<td></td>
<td>Debug tool setting: Little endian</td>
</tr>
<tr>
<td>iodefine.h</td>
<td>Version 2.2</td>
</tr>
<tr>
<td>Sample program</td>
<td>Version 1.00</td>
</tr>
<tr>
<td>Evaluation Board</td>
<td>Renesas Electronics: Renesas RX65N Cloud Kit</td>
</tr>
<tr>
<td></td>
<td>(RTK5RX65Nxxxxxxxxxxx)</td>
</tr>
<tr>
<td></td>
<td>• On-board MCU: See above.</td>
</tr>
<tr>
<td></td>
<td>• Power supply: Power supplied via USB.</td>
</tr>
<tr>
<td></td>
<td>• Debug LCDs connected.</td>
</tr>
<tr>
<td></td>
<td>• Wi-Fi module connected.</td>
</tr>
<tr>
<td>LCD module</td>
<td>ST7735</td>
</tr>
<tr>
<td>Wi-Fi module</td>
<td>SX-ULPGN</td>
</tr>
<tr>
<td>Accelerometer module</td>
<td>ADXL335</td>
</tr>
</tbody>
</table>
2. File Structure Associated with This Application Note

2.1 File Structure Associated with This Application Note

Figure 2.1 shows the file structure associated with this application note. When the contents of the ZIP file in which this application note is distributed is unzipped, a folder is created with the same name as the ZIP file. The “workspace_Cloud_Kit_demo_example” folder within this folder contains an e² studio workspace that includes “amazon-freertos-rx-1.0.3” a project in e² studio format. As shown in Figure 2.2, the project folder contains the sample program source code files as well as e² studio configuration files and this application note. Figure 2.2 shows the configuration viewed from File Explorer.

![Diagram](image_url)

**Figure 2.1** File Structure Associated with This Application Note
Figure 2.2  Folder Structure of Sample Project
2.2 Structure of Sample Program

Figure 2.3 shows the structure of the sample program, and Table 2.1 and Table 2.2 list the software modules used. The FIT modules and DSP library can be obtained from the Renesas website. The FreeRTOS module can be obtained from GitHub. Driver software for the other peripheral functions is generated by using the Code Generator function of e² studio. For details of each software module, refer to the associated application note or the e² studio help system.

![Figure 2.3 Structure of Sample Program](image)
### Table 2.1  List of Software Modules Used (Renesas)

<table>
<thead>
<tr>
<th>Module</th>
<th>Document Title</th>
<th>Document No.</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td>—</td>
<td>—</td>
<td>Module containing the main function developed for the program described in this application note</td>
</tr>
<tr>
<td>r_dsp</td>
<td>—</td>
<td>—</td>
<td>DSP library operation module developed for the program described in this application note</td>
</tr>
<tr>
<td>r_lcd_driver</td>
<td>—</td>
<td>—</td>
<td>Debug LCD control module developed for the program described in this application note</td>
</tr>
<tr>
<td>BSP</td>
<td>RX Family Board Support Package Module Using Firmware Integration Technology</td>
<td>R01AN1685</td>
<td>FIT module</td>
</tr>
<tr>
<td>DMAC</td>
<td>RX Family DMAC Module Using Firmware Integration Technology</td>
<td>R01AN2063</td>
<td>FIT module</td>
</tr>
<tr>
<td>SCI</td>
<td>RX Family SCI Module Using Firmware Integration Technology</td>
<td>R01AN1815</td>
<td>FIT module</td>
</tr>
<tr>
<td>RSPI</td>
<td>RX Family RSPI Module Using Firmware Integration Technology</td>
<td>R01AN1827</td>
<td>FIT module</td>
</tr>
<tr>
<td>BYTEQ</td>
<td>RX Family BYTEQ Module Using Firmware Integration Technology</td>
<td>R01AN1683</td>
<td>FIT module</td>
</tr>
<tr>
<td>CMT</td>
<td>RX Family CMT Module Using Firmware Integration Technology</td>
<td>R01AN1856</td>
<td>FIT module</td>
</tr>
<tr>
<td>S12AD</td>
<td>—</td>
<td>—</td>
<td>Driver function generated by Code Generator</td>
</tr>
<tr>
<td>CMT</td>
<td>—</td>
<td>—</td>
<td>Driver function generated by Code Generator</td>
</tr>
<tr>
<td>ELC</td>
<td>—</td>
<td>—</td>
<td>Driver function generated by Code Generator</td>
</tr>
<tr>
<td>RX DSP library</td>
<td>RX Family DSP Library Version 5.0</td>
<td>R01AN4359</td>
<td>DSP library</td>
</tr>
</tbody>
</table>

### Table 2.2  List of Software Modules Used (FreeRTOS)

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FreeRTOS Kernel</td>
<td>Realtime operating system</td>
</tr>
<tr>
<td>MQTT Agent</td>
<td>Open source daemon for managing MQTT libraries</td>
</tr>
<tr>
<td>WiFi Mgmt. Library</td>
<td>Library providing APIs for managing Wi-Fi devices</td>
</tr>
<tr>
<td>FreeRTOS Internal Libraries</td>
<td>Libraries providing added functionality for FreeRTOS Kernel and its internal libraries</td>
</tr>
</tbody>
</table>
3. Running the Sample Program

The procedure for running the program described in this application note is shown below.

3.1 Importing the Project

After unzipping the zip file of the sample project attached to this application note, move the project (amazonfreertos-rx-1.0.3) to the root folder (C:\).* Next, launch e² studio, and when the Eclipse launcher is displayed, specify a folder for the workspace (in this document, C:\workspace).

When e² studio starts, refer to Figure 3.2 and import the project. When the import is completed, the project “aws_demos” will be displayed in “Project Explorer.”

Note: 1. If you move the project to the workspace folder, an error may occur when you attempt to build it.

![Eclipse Launcher](image)

**Figure 3.1 Eclipse Launcher**
Start the e² studio, and select menu [File] >> [Import…].

Select [Existing Projects into Workspace].

Specify the directory which stored the project to import.
C:\amazon-freertos-rx-1.0.3\projects\renesas\rx65n-cloud-kit-uart-sx-ulpg\e2studio\aws_demos

Select [Select root directory].

Figure 3.2 Importing the Project into e² studio
3.2 Connecting Equipment

Connect the necessary equipment. Figure 3.3 shows how to connect each device. Refer to Table 3.1 for the connections between the accelerometer and RX65N Cloud Kit.

**Note:**
1. Reference: Analog Devices EVAL-ADXL335

![Figure 3.3 Equipment Connections](image)

**Table 3.1 Pin Connections**

<table>
<thead>
<tr>
<th>Number</th>
<th>Accelerometer Module</th>
<th>Cloud Kit</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>Vdd</td>
<td>TARGET_VCC</td>
</tr>
<tr>
<td>[2]</td>
<td>GROUND</td>
<td>GROUND</td>
</tr>
<tr>
<td>[3]</td>
<td>x-axis output</td>
<td>AN000</td>
</tr>
<tr>
<td>[4]</td>
<td>y-axis output</td>
<td>AN001</td>
</tr>
<tr>
<td>[5]</td>
<td>z-axis output</td>
<td>AN002</td>
</tr>
</tbody>
</table>
3.2.1 Hardware Configuration
Table 3.2 shows hardware configuration of the demo system.

<table>
<thead>
<tr>
<th>Item</th>
<th>Product name, model name</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Board for RX65N</td>
<td>RTK5RX65Nxxxxxxxxxxxx</td>
<td>Accessory of Renesas RX65N Cloud Kit</td>
</tr>
<tr>
<td>Cloud Option Board</td>
<td>RTK5RX65Nxxxxxxxxxxxx</td>
<td>Accessory of Renesas RX65N Cloud Kit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rework is needed (Figure 3.5)</td>
</tr>
<tr>
<td>LCD Module</td>
<td>---</td>
<td>Accessory of RSK+RX65N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Demo can be run without LCD</td>
</tr>
<tr>
<td>Extension Board</td>
<td>Pmod TPH2</td>
<td>Rework is needed (Figure 3.4)</td>
</tr>
<tr>
<td>WiFi Module</td>
<td>SX-ULPGN</td>
<td>Accessory of Renesas RX65N Cloud Kit</td>
</tr>
<tr>
<td>Accelerometer Module</td>
<td>ADXL-335</td>
<td>Reference: Analog Devices EVAL-ADXL335</td>
</tr>
</tbody>
</table>
3.2.2 Hardware Rework

Refer to Figure 3.4 and Figure 3.5, rework hardware of the demo system.

![Figure 3.4 Rework of Extension Board](image1)

![Figure 3.5 Rework of Cloud Option Board](image2)
3.3 Network Environment and Certificate Issuance

To connect to AWS, you will need information (SSID and password) for the access point used to connect to the network and a certificate issued by AWS. After the certificate is initially issued by AWS, you will need to update the sample program with the access point information and certificate information. Depending on the version, the details of the AWS Management Console page may differ somewhat. If the page you see differs from the one shown in this application note, follow the instructions you see displayed on the page.

The procedure is as follows.

(1) Logging In to AWS

Log in to the AWS Management Console from the AWS homepage. Enter the URL https://aws.amazon.com/ to access the homepage. From the homepage, click the link shown in Figure 3.6. The account ID and password input page appears. Log in to your registered AWS account.

(2) Accessing IoT Core

After you have logged in, access IoT Core as shown in Figure 3.7.
### (3) Creating a Security Policy

After accessing IoT Core, on the menu on the left click Security → Policy → Create Policy. On the page that appears next input the policy name to be registered and settings. Figure 3.8 shows the policy creation procedure and Figure 3.9 shows the settings.

![Figure 3.8 Policy Setting Procedure](image)

![Figure 3.9 Policy Setting Details](image)

```json
afs_handson_test_02
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "*"
    }
  ]
}
```

Defines access permissions for RX65N Cloud Kit.
- iot:Connect
  - Connect to AWS IoT.
- iot:Publish
  - Publish (transmit) topics.
- iot:Subscribe
  - Subscribe to (receive) topics.
- iot:Receive
  - Receive message from AWS IoT.
(4) Registering a Thing and Issuing a Certificate

On the menu on the left of the IoT Core page click Manage → Thing → Register Thing (Figure 3.10). After you input the name of the thing (Figure 3.11), create a certificate and download it (Figure 3.12 and Figure 3.13). Finally, attach the policy to link the policy and the thing (Figure 3.14 and Figure 3.15). Do not fail to download the newly issued certificate from this page. You will not be able to download the public key and private key again after this point. Also, make a note in a text editor of the name of the thing specified here because it will be used again later when setting up the sample program.

![Figure 3.10 Thing Registration Page](image1)

![Figure 3.11 Thing Registration Setting Page](image2)

![Figure 3.12 Issuing a Certificate](image3)
In order to connect a device, you need to download the following:

- A certificate for this thing
- A public key
- A private key

You also need to download a root CA for AWS IoT:

- A root CA for AWS IoT

**Figure 3.13  Downloading a Certificate**

**Figure 3.14  Attaching a Policy**

**Figure 3.15  Selecting a Policy**
(5) Confirming the IoT Endpoint

The IoT endpoint will differ depending on the region indicated in the upper right of the AWS Management Console page. Click Settings on the menu on the left of the IoT Core page to confirm the endpoint (Figure 3.16). Make a note of the endpoint in a text editor because it will be used again later when setting up the sample program.

![Figure 3.16 Confirming the Endpoint](image)

(6) Macro Settings

Enter the newly created thing name and the SSID and password settings of the access point to be used for communication in aws_clientcredential.h. Figure 3.17 shows the locations of the various settings.

![Figure 3.17 Macro Settings](image)
(7) Importing the Certificate and Private Key

Use the certificate and private key to create a header file (Figure 3.18). Then overwrite the existing header file with the newly created header file (Figure 3.19). The information in the header file and the registered thing information make it possible to communicate with AWS.

![Figure 3.18  Creating a Header File](image)

![Specify Certificate and Private Key files downloaded at "(4) Registering a Thing and Issuing a Certificate"
Certificate: `*-certificate.pem.crt`
Private Key: `*-private.pem.key`](image)

![Figure 3.19  Overwriting the Header File](image)

![Overwrite](image)
3.4 Elasticsearch Service Domain Creation

In order to visualize data using the Elasticsearch Service, you must first create a domain. Creating a domain on the Elasticsearch Service generates a fee based on the amount of time the domain is in existence. Please make sure to delete your domain after the demo finishes. Refer to 5, Important Note, for details. The details of the Management Console page may differ somewhat depending on the version. If the page you see differs from the one shown in this application note, follow the instructions you see displayed on the page.

The Elasticsearch Service domain creation procedure is as follows.

(1) Accessing the Elasticsearch Service
From the top page of the Management Console, access the Elasticsearch Service as shown in Figure 3.20.

(2) Domain Settings
In order to use the Elasticsearch Service, you must create a domain. First of all, click the Create New Domain button shown in Figure 3.21. After you click the button the domain settings page appears. Enter settings as shown in Figure 3.22, Figure 3.23, and Figure 3.24. Specify a domain name of your choice, making sure to follow the rules regarding usable characters. After all the settings have been entered, confirm the setting details as shown in Figure 3.25 to complete the process.

Figure 3.20 Accessing the Elasticsearch Service

Figure 3.21 Creating a Domain
Figure 3.22  Domain Creation Page (1)

Create Elasticsearch domain

Choose deployment type

1. Development and testing
   - Use Availability Zones for testing and development.

Version

2. Select the version of Elasticsearch for your domain.

Choose setup

3. Elasticsearch domain name: afs-handson-test
   - A domain name can include a maximum of 24 characters, and must be between 3 and 24 characters.
   - Valid characters are a-z (lowercase only), 0-9, and - (hyphen).

Data nodes

4. Instance type: t2.small.elasticsearch
   - The selected instance type (t2.small.elasticsearch) does not support encryption at rest.

Number of nodes

5. 1

Scroll down.

Figure 3.23  Domain Creation Page (2)
Figure 3.24 Domain Creation Page (3)
Figure 3.25  Confirm Settings Page
(3) Completing Domain Creation

After you complete the domain creation procedure, you may need to wait up to 10 minutes before creation of the domain is finalized. Wait until the domain status is shown as Active, as shown in Figure 3.26, before proceeding. If the domain status does not change after waiting for a while, try clicking the Refresh button in the upper right of the page. Once the status is shown as Active, click the Kibana link.

![Figure 3.26 New Domain Confirmation Page](image)
(4) Creating an Index on Kibana

When you click the Kibana link, a page like that shown in Figure 3.27 appears. Click the items indicated in Figure 3.27 to proceed. When you click the Get to work button, the page shown in Figure 3.28 appears. Input commands to the Console as shown in Figure 3.28. Click the button marked [5] in the figure, and the response appears in the right pane of the Console.

(5) Accessing IoT Core

To display data using the Elasticsearch Service, you must first set up IoT Core so that data sent to AWS is then sent to the Elasticsearch Service. Return to the top page and access IoT Core.
(6) Creating IoT Rules

IoT rules enable you to specify that data sent from Cloud Kit is sent to the Elasticsearch Service domain you created. Access ACT from the IoT Core page, as shown in Figure 3.29, and then enter settings as shown in Figure 3.30, Figure 3.31, Figure 3.32, Figure 3.33, and Figure 3.34.

![Figure 3.29 Accessing ACT](image)

![Figure 3.30 Creating IoT Rules (1)](image)

---

**Copy & paste this text into “Rule query statement”**.

**After entering the text, press the Enter key!**
Figure 3.31 Creating IoT Rules (2)
Figure 3.32  Creating IoT Rules (3)
**Figure 3.33  Creating IoT Rules (4)**

**Figure 3.34  Creating IoT Rules (5)**
3.5 Running the Sample Program and Confirming Operation

Follow the steps below to connect the debug tool and run the sample program.

1) Build the sample program in e² studio and download it to the MCU.
2) From e² studio, reset the MCU and run the sample program.
3) Execution of main.c stops at line vTaskDelay(10000);, so click the Resume button to resume execution.
4) After execution is resumed, the demo starts.*¹

Note: 1. If the Wi-Fi status shown on the LCD is DISCONNECTED, connecting to Wi-Fi failed. In this case, quit the program, confirm that the details shown in (6) Macro Settings in 3.3, Network Environment and Certificate Issuance, are correct, and then retry the above procedure from step 1).

3.5.1 LCD Display Details

When you run the sample program, the processing results are displayed on the LCD module as shown in Figure 3.35. The display on the LCD module is refreshed in regular cycles (of about one second).

![Figure 3.35 LCD Module Display Results](image-url)
3.5.2 Viewing Output Log in Tera Term
After you run the sample program, you can view the communication log with AWS in Tera Term. Launch Tera Term, and then make settings as shown in Figure 3.36. Launch Tera Term and select **Setup → Serial port** to open the window shown in Figure 3.37. Enter a baud rate setting of **115200** here. Once the setting has been entered correctly, a log such as that shown in Figure 3.38 is output to the screen.

![Figure 3.36 Tera Term Startup Screen](image1)

![Figure 3.37 Baud Rate Setting](image2)
Figure 3.38 Communication Log Displayed in Tera Term
3.5.3 Displaying Data Using the Elasticsearch Service

You can send the FFT results to AWS via Wi-Fi and use the Elasticsearch Service to visualize the data in graphical form. The procedure to follow is described below. If the data is transmitted correctly, a graph like that shown in Figure 3.45 will be displayed.

(1) Going to the Graphical Display Page

Access the Kibana page and select Index Patterns from the menu on the left as shown in Figure 3.39. After you make the selection, a Create index pattern page such as that shown in Figure 3.40 appears. Make settings as shown in Figure 3.40. Next, click Line in the menu on the left, as shown in Figure 3.41. Then proceed as shown in Figure 3.42 to go to the graphical display page.

Notes:
1. Depending on the version of Kibana, press the Discover button instead of the Management button.
2. If you press Management button, please press the Create index pattern button.

Figure 3.39  Accessing Index Patterns from the Menu on the Left of the Kibana Page

Figure 3.40  Creating an Index Pattern
Figure 3.41 Accessing the Line Visualization

Figure 3.42 Graphical Display Page
(2) Displaying a Graph

Set the x-axis and y-axis to display a graph. First, set the display update frequency as shown in Figure 3.43. The demo sends data once every second, so the update frequency should be 1 second. Next, specify the x-axis label (time) and y-axis label (peak frequency or peak frequency magnitude) to be displayed, as shown in Figure 3.44. To display the peak frequency, select x.Freq, y.Freq, and z.Freq, and to display the peak frequency magnitude, select x.Mag, y.Mag, and z.Mag.

![Figure 3.43 Graph Display Update Frequency]
Figure 3.44  Graph Axis Label Settings
Figure 3.45 Graph Display Result
### 3.6 User-Modifiable Settings

Table 3.3 and Table 3.4 list system settings that can be modified by the user. The settings listed in Table 3.3 are defined in the main.h file. The settings listed in Table 3.4 are defined in the aws_clientcredential.h file. Note that the settings listed in Table 3.4 are used to connect to Wi-Fi and AWS. For details, refer to 3.3, Network Environment and Certificate Issuance.

#### Table 3.3 Modifiable Settings (renesas_demo_data_uploader.h)

<table>
<thead>
<tr>
<th>Function/Definition Name</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS_RESULT_JUDGEMENT</td>
<td>For x, y and z-axis, sets the expected peak frequency value (frequency at maximum magnitude according to FFT processing result). Setting unit: Hz. Setting range: 1 to 499.</td>
<td>150</td>
</tr>
<tr>
<td>EXPECTED_FREQUENCY_X</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>EXPECTED_FREQUENCY_Y</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>EXPECTED_FREQUENCY_Z</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>EXPECTED_MAGNITUDE_X</td>
<td>For x, y and z-axis, sets the pass (OK) judgement threshold for magnitude at the peak frequency specified by EXPECTED_FREQUENCY. Setting range: 1 to 2147483647.</td>
<td>100000000</td>
</tr>
<tr>
<td>EXPECTED_MAGNITUDE_Y</td>
<td></td>
<td>100000000</td>
</tr>
<tr>
<td>EXPECTED_MAGNITUDE_Z</td>
<td></td>
<td>100000000</td>
</tr>
</tbody>
</table>

**Communication quality**

<table>
<thead>
<tr>
<th>Function/Definition Name</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>QOS</td>
<td>Sets the communication quality for communication with MQTT. Setting values: QoS0 (data transmitted only once) QoS1 (data always transmitted at least one time)</td>
<td>QoS0</td>
</tr>
</tbody>
</table>

#### Table 3.4 Modifiable Settings (aws_clientcredential.h)

<table>
<thead>
<tr>
<th>Function/Definition Name</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientcredentialWIFI_SSID</td>
<td>Input the SSID of the Wi-Fi device to connect to. Choose the setting that matches your network environment.</td>
<td>Paste Wi-Fi SSID here.</td>
</tr>
<tr>
<td>clientcredentialWIFI_PASSWORD</td>
<td>Input the password of the Wi-Fi device to connect to. Choose the setting that matches your network environment.</td>
<td>Paste Wi-Fi password here.</td>
</tr>
<tr>
<td>clientcredentialMQTT_BROKER_ENDPOINT[]</td>
<td>Input the AWS endpoint to connect to. The correct setting varies by region, so choose the setting that matches your region.</td>
<td>Paste AWS IoT Broker endpoint here.</td>
</tr>
<tr>
<td>clientcredentialIOT_THING_NAME</td>
<td>Input the name of the thing for which to connect to AWS. Use the name of the thing for which you created a certificate as described in 3.3, Network Environment and Certificate Issuance.</td>
<td>Paste AWS IoT Thing name here.</td>
</tr>
</tbody>
</table>
4. Description of Sample Program

4.1 Overview of Sample Program

Table 4.1 lists the processing performed by the sample program.

Table 4.1 Roles of Processing Tasks

<table>
<thead>
<tr>
<th>Processing</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>main processing</td>
<td>• Initializes peripheral modules and DSP-related processing.</td>
</tr>
<tr>
<td></td>
<td>• Starts the first DMA transfer.</td>
</tr>
<tr>
<td></td>
<td>• Notifies DMA transfer end interrupt processing of next DMA destination address</td>
</tr>
<tr>
<td></td>
<td>• Performs normalization processing of input data.</td>
</tr>
<tr>
<td></td>
<td>• Performs DSP-related processing such as FFT processing.</td>
</tr>
<tr>
<td></td>
<td>• Judges FFT processing results and updates LCD module display.</td>
</tr>
<tr>
<td>DMA transfer end interrupt</td>
<td>Makes settings for second and subsequent DMA transfers and starts</td>
</tr>
<tr>
<td>processing</td>
<td>transfer.</td>
</tr>
<tr>
<td>CMT2 interrupt processing</td>
<td>Redraws the LCD module display (controlled by the r_lcd_driver module).</td>
</tr>
<tr>
<td>AWS transmission processing</td>
<td>Performs processing to transmit peak frequency and peak frequency</td>
</tr>
<tr>
<td></td>
<td>magnitude FFT processing results.</td>
</tr>
</tbody>
</table>
4.2 Processing Sequence

The processing sequence of the sample program consists of two parts that operate independently: on the one hand the main processing, the DMA transfer end interrupt processing, and data transmission processing, and on the other the CMT2 interrupt processing. Figure 4.1 shows the sequence for the main processing, DMA transfer end interrupt processing, and data transmission processing, and Figure 4.2 shows the sequence for the CMT2 interrupt processing.

As shown in the figure, the first DMA transfer is enabled when A/D conversion ends. When DMA transfer of the specified number of data units finishes, a DMA transfer-end interrupt request occurs. This triggers execution of the DMA transfer end interrupt processing and main processing routine, in that order. When DSP finishes, the peak frequency and peak frequency magnitude values are sent to the queue, and processing branches to the task for sending data to AWS. When sending of data to AWS finishes, the queue is emptied and processing transitions to the task for DSP. Thereafter, the same processing sequence is executed repeatedly.
After initializing the peripheral function, the main processing uses CMT2 and the RSPI to process displaying indications on the LCD module. The CMT2 interrupt processing uses the RSPI to send one line of data to the LCD module. The display is updated, one line at a time, each time a CMT2 interrupt request occurs. The height of the LCD module is 128 pixels, so it takes 128 CMT2 interrupt requests to update the entire screen.
4.3 Processing Flowchart

Figure 4.3 shows the processing flow of the sample program. The DSP for only one input signal is shown, but in fact three input signals are processed.

![Processing Flowchart Diagram]

**Figure 4.3 Processing Flow**
4.4 Details

4.4.1 Normalization Processing

The sample program performs the normalization processing (bias processing and scaling) shown in Figure 4.4.

- Normalization processing
  The data input by the S12AD is in 12-bit (unsigned) format, so it must be normalized to 31-bit (signed) format in order to obtain adequate operation results from FFT processing. The normalization processing consists of bias processing and scaling.
4.4.2 FFT Processing

The sample program uses the DSP library API functions shown in Figure 4.5 to output the frequency spectrum magnitude characteristics of the input signal.

- **APIs used for FFT processing**
  FFT processing consists of the real FFT operation function `R_DSP_FFT_i32ci32` and the complex number magnitude function `R_DSP_VecCplxMag_ci32i32`. `R_DSP_FFT_i32ci32` outputs FFT results as complex number.
  In order to get magnitudes of the frequency spectrum on input signals, the sample program converts the complex number outputs of `R_DSP_FFT_i32ci32` into integer number of magnitudes with `R_DSP_VecCplxMag_ci32i32`.

- **Input and output data**
  FFT processing outputs N/2 counts of results for N counts of inputs. The N corresponds to the FFT point count and a unit of executing `R_DSP_FFT_i32ci32`. The point count of results is 1/2 of the input in compliance with the specification of `R_DSP_FFT_i32ci32`. The sample program inputs data for 1,024 points and outputs results for 512 points.
  For details, refer to the RX DSP Library APIs User’s Manual: Software.

- **Output data corresponding to frequencies**
  The outputs of the FFT processing shows magnitudes corresponding to frequencies. As shown in Table 4.2, the magnitudes corresponding to frequencies are stored in the output buffer.
  The frequency interval is calculated with the sampling frequency and the FFT point count. In the sample program, it equivalents approximately 0.97 Hz by the following calculation.

\[
\text{Sampling Frequency} / \text{FFT Point Count} = \frac{1000}{1024} = 0.9765625 \text{ Hz}
\]

### Table 4.2 Input Data corresponding to Frequencies

<table>
<thead>
<tr>
<th>Data No.</th>
<th>Frequency [Hz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.97... x 0 = 0</td>
</tr>
<tr>
<td>1</td>
<td>0.97... x 1 = 0.97</td>
</tr>
<tr>
<td>2</td>
<td>0.97... x 2 = 1.95</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>510</td>
<td>0.97... x 510 = 498.04</td>
</tr>
<tr>
<td>511</td>
<td>0.97... x 511 = 499.02</td>
</tr>
</tbody>
</table>
4.4.3 Judgement of FFT Processing Results
A judgement is made based on the FFT processing results stored in the output buffer. The judgement is "pass" (OK) if the following two conditions are met by x-axis, y-axis and z-axis processing results:

- Peak frequency = expected frequency (definition EXPECTED_FREQUENCY)
- Peak frequency magnitude ≥ expected magnitude value (definition EXPECTED_MAGNITUDE)

The peak frequency is the frequency with the maximum magnitude according to the FFT processing results. The expected frequency is set in increments of 1 Hz, but the FFT processing output data and frequency are delineated in increments of approximately 0.97 Hz, as indicated in Table 4.2. Therefore, the expected frequency is converted to the corresponding data number to perform the comparison. The acceptable range is specified as extending one data number before and after the converted data number.

4.4.4 Data Transmission to AWS
The FFT processing results are sent to AWS. Data transmission to AWS takes place via Wi-Fi. If communication with AWS cannot be established, data transmission processing does not occur and only updating of the display on the LCD module occurs. Communication using MQTT is employed for data transmission.

The demo sends the peak frequency and peak frequency magnitude from Cloud Kit, and the peak frequency and peak frequency magnitude are received by AWS.

4.4.5 Viewing Communication Log Output in Tera Term
The AWS communication log is displayed. Also, an error is displayed if communication with AWS cannot be established due to a problem involving the Wi-Fi or AWS. A message at the beginning of the log indicates what type of processing the log is recording. Table 4.3 lists the types of processing. Since data transmission is not possible when communication with Wi-Fi or AWS is interrupted, "Communication is disconnected" is displayed at one-second intervals in this case.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tmr Svc</td>
<td>Task generation and initialization processing</td>
</tr>
<tr>
<td>MQTT</td>
<td>MQTT processing to receive transmitted data</td>
</tr>
<tr>
<td>Sensor Upload</td>
<td>Processing to transmit data to AWS</td>
</tr>
</tbody>
</table>
4.4.6 Processing to Update Display on LCD Module
Table 4.4 lists the information displayed on the LCD module. Processing to change the display is divided between the following two processing routines. The FFT processing results, the judgement results based on processing values and expected values, and the connection status with AWS are displayed. The displayed content is updated at one-second intervals. Since there are FFT processing results for three axes, the peak frequency and peak frequency magnitude value are displayed for three axes.

- **DSP task**
  The FFT processing results, the judgement results based on processing values and expected values, and the connection status with AWS are converted into data for display and written to the display update buffer.

- **CMT2 interrupt processing**
  Reads the display data from the display update buffer and uses it to update the LCD module, one line at a time each time a CMT2 interrupt occurs, as described in 3.2, Connecting Equipment.

<table>
<thead>
<tr>
<th>Table 4.4 Information Displayed on LCD Module</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item</strong></td>
</tr>
<tr>
<td>Peak frequency</td>
</tr>
<tr>
<td>Peak frequency magnitude value</td>
</tr>
<tr>
<td>Judgement</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Wi-Fi connection status</td>
</tr>
<tr>
<td>AWS connection status</td>
</tr>
</tbody>
</table>

4.4.7 QoS Setting
QoS stands for “quality of service,” and it indicates the communication quality for communication with MQTT.

Table 4.5 shows the setting values. The demo uses the default value of QoS0 to speed up communication. If more accurate data communication is necessary, change the value to QoS1.

<table>
<thead>
<tr>
<th>Table 4.5 QoS Setting Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setting Value</strong></td>
</tr>
<tr>
<td>QoS0</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>QoS1</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
4.5 File Structure

Table 4.6 shows the file structure of the modules described in this application note. Table 4.7 to Table 4.10 list the variables used in the source files. For details of the other modules, refer to their respective application notes.

Table 4.6 File Structure of Modules

<table>
<thead>
<tr>
<th>Module/File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>Main processing of sample program</td>
</tr>
<tr>
<td>renesas_demo_data_uploader.c</td>
<td>Main processing, DMA transfer end interrupt processing, etc.</td>
</tr>
<tr>
<td>renesas_demo_data_uploader.h</td>
<td>renesas_demo_data_uploader.c header file</td>
</tr>
<tr>
<td>r_dsp</td>
<td>DSP-related processing</td>
</tr>
<tr>
<td>r_normalize.c</td>
<td>Normalization processing</td>
</tr>
<tr>
<td>r_normalize.h</td>
<td>r_normalize.c header file</td>
</tr>
<tr>
<td>r_dsp_real_fft.c</td>
<td>FFT processing initialization, FFT processing</td>
</tr>
<tr>
<td>r_dsp_real_fft.h</td>
<td>r_dsp_real_fft.c header file</td>
</tr>
<tr>
<td>r_lcd_driver</td>
<td>LCD module control processing</td>
</tr>
<tr>
<td>r_ascii.c</td>
<td>Character table</td>
</tr>
<tr>
<td>r_ascii.h</td>
<td>r_ascii.c header file</td>
</tr>
<tr>
<td>r_lcd_driver.c</td>
<td>LCD module control processing</td>
</tr>
<tr>
<td>r_lcd_driver_private.h</td>
<td>Private header file</td>
</tr>
<tr>
<td>r_lcd_driver_if.h</td>
<td>Interface header file</td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| dsp_analysis_and_judge_task | • Initializes peripheral devices.  
• Starts first DMA transfer.  
• Notifies next DMA transfer destination to DMA transfer end interrupt processing.  
• Performs DSP.  
• Displays judgement of processing results on LCD module. |
| set_buf_info              | Sets the next DMA transfer destination address and data count in the variables used for communication between the main routine and the DMA transfer end interrupt processing.                                    |
| init_dmac_s12ad           | • Initializes DMAC channel 0 with S12AD channel 0 as the transfer source.  
• Initializes DMAC channel 1 with S12AD channel 1 as the transfer source.  
• Initializes DMAC channel 2 with S12AD channel 2 as the transfer source. |
| callback_dmac_s12ad_ch0    | The callback function registered in the DMAC module by the main processing routine  
• DMAC channel 0 DMA transfer end interrupt processing  
• Sets the next transfer destination in the DMAC channel 0 and enables the DMA transfer.  
• Requests the next transfer destination information from the main processing. |
| callback_dmac_s12ad_ch1    | The callback function registered in the DMAC module by the main processing routine  
• DMAC channel 1 DMA transfer end interrupt processing  
• Sets the next transfer destination in the DMAC channel 1 and enables the DMA transfer.  
Requests the next transfer destination information from the main processing. |
| callback_dmac_s12ad_ch2    | The callback function registered in the DMAC module by the main processing routine  
• DMAC channel 2 DMA transfer end interrupt processing  
• Sets the next transfer destination in the DMAC channel 2 and enables the DMA transfer.  
Requests the next transfer destination information from the main processing. |
| init_lcd_display          | Initializes the LCD module and sets the initial display information.                                                                                                                                        |
| judge_and_update          | Makes a judgement based on the FFT results and updates the information displayed on the LCD module.                                                                                                          |
### Table 4.8  r_normalize.c File Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_Normalize_Operation</td>
<td>Normalization processing</td>
</tr>
</tbody>
</table>

### Table 4.9  r_dsp_real_fft.c File Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_DSP_REAL_FFT_Init</td>
<td>Initializes FFT processing.</td>
</tr>
<tr>
<td>R_DSP_REAL_FFT_Operation</td>
<td>Executes FFT processing.</td>
</tr>
</tbody>
</table>

### Table 4.10  r_lcd_driver.c File Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_LCD_Driver_Open</td>
<td>Initializes the LCD module and driver.</td>
</tr>
<tr>
<td>R_LCD_Driver_PrintString</td>
<td>Writes character strings to the update buffer.</td>
</tr>
<tr>
<td>R_LCD_Driver_PeriodicCallback</td>
<td>Processing called from the CMT2 interrupt processing to update one line of the LCD module.</td>
</tr>
</tbody>
</table>
5. Important Note

5.1 Frequency Value of FFT Processing Result

The RX65N on the RX65N Cloud Kit operates with HOCO as the clock source. HOCO includes an error of ±3%, which is the error in the sampling frequency of the A/D conversion processing of the sample program. Since the sampling frequency error becomes the frequency error of the FFT processing result, the peak frequency display of the demo system may be shifted by a few Hz from the frequency of the input signal. When applying the demo system to a system that requires more accurate frequency analysis, use a highly accurate oscillator as the RX65N clock source and change the clock settings of the sample program.

Example: Using a 24 MHz crystal oscillator

To use an oscillator, make changes to r_bsp_config.h as shown in Figure 5.1. If the oscillator generates a frequency other than 24 MHz, make adjustments such that the PLL circuit output is 240 MHz.

![Figure 5.1 Settings in r_bsp_config.h](image)

5.2 Warning Message at Project Building

When building the sample program attached to this document, warnings related to FreeRTOS coding style are output, but this does not affect the operation of the demo system.
5.3 A Fee for the Elasticsearch Service

Creating a domain on the Elasticsearch Service generates a fee while the domain is in existence. After the demo finishes, make sure to delete your domain as shown in Figure 5.2.

Figure 5.2 Domain Deletion Procedure
6. Reference

6.1 System Operation Monitoring in e² studio
Using the Waveform rendering function of e² studio, the signal input to the RX65N and the FFT processing result can be monitored.

![Figure 6.1 Waveform Rendering Display Example (Data Stored in the Input Buffer)](image1)

![Figure 6.2 Waveform Rendering Display Example (Frequency Magnitude Characteristics by FFT Processing)](image2)

First, make a debug connection with the RX65N Cloud Kit, display the memory view according to the procedure shown in Figure 6.3, and specify the buffer to be displayed.

Next, display the waveform of the input signal and the frequency magnitude characteristics in a graph by the procedure shown in Figure 6.4.
Figure 6.3 Memory View Display Procedure

“Memory”

Specify “gs_input_buffer_x” for input buffer for x-axis
Specify “gs_output_buffer_x” for FFT processing result (Frequency-magnitude characteristics)

“Add Memory Monitor”

Memory View displays values of data stored in gs_input_buffer_x.
Configure parameters such as data size in “Waveform Properties”. Specify 16bit for “gs_input_buffer_x” and 32bit for “gs_output_buffer_x”.

“Y-axis settings” is corresponding to vertical axis and “Buffer Size” is corresponding to horizontal axis. Adjust the parameters according to the data to display.

Press “OK” to display a waveform of “gs_input_buffer_x”.

Press “Real-time Refresh” button to update a waveform automatically.

Figure 6.4 Waveform Rendering Display Procedure
6.2 Memory Usage

Table 6.1 lists the memory usage of the sample program.

The figures for ROM and RAM were calculated based on a .map file generated under the conditions listed in Table 1.1. The figures for user stack and interrupt stack were obtained by measuring the actual stack memory usage while the sample program was running.

<table>
<thead>
<tr>
<th>Item</th>
<th>Measured Value [Bytes]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td>381882</td>
<td>Sample program ROM usage</td>
</tr>
<tr>
<td>RAM</td>
<td>276570</td>
<td>Sample program RAM usage</td>
</tr>
<tr>
<td>User stack</td>
<td>6732</td>
<td>Sample program stack memory usage</td>
</tr>
<tr>
<td>Interrupt stack</td>
<td>117</td>
<td></td>
</tr>
</tbody>
</table>
7. Obtaining the Development Environment

7.1 Obtaining e² studio
Download e² studio from the following webpage:


This document assumes that 2021-01 or later of e² studio is used. If a version earlier than 2021-01 is used, some e² studio functions may not be supported. Make sure to download the latest version of e² studio on the website.

7.2 Obtaining the Compiler Package
Download the RX Family C/C++ Compiler Package from the following webpage:


8. Supplement

8.1 Notes on Using the Free Evaluation Version of the RX Family C/C++ Compiler Package
The free evaluation version of the RX Family C/C++ Compiler Package can be used only for a limited period of time and is subject to functional limitations as well. Once the trial usage period has expired, the link size is limited to 128 KB or less, and this may prevent the load module from being generated correctly.

For details, refer to the Evaluation Software Tools page on the Renesas website.


8.2 DSP Library for RX Family
The DSP library is applied for the DSP processing such as FFT of this sample program.

For more information and downloads of the DSP library, refer to the DSP Library for RX Family page on the Renesas website.

https://www.renesas.com/software-tool/dsp-library-rx-family

9. Reference Documents

- RX Family Board Support Package Module Using Firmware Integration Technology (R01AN1685)
- Renesas e² studio Smart Configurator User Guide (R20AN0451)
- RX65N Group, RX651 Group User’s Manual: Hardware (R01UH0590)

The latest version can be downloaded from the Renesas Electronics website.
## Revision History

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Description</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Apr. 23, 2021</td>
<td>—</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>
General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between $V_{IL}$ (Max.) and $V_{IH}$ (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between $V_{IL}$ (Max.) and $V_{IH}$ (Min.).

7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system- evaluation test for the given product.
1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.

2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.

3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.

5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alterations.

6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space systems; underwater repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product.

   RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVIOLABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.

12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.

13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.