Aeropoint GUI

User's Manual

CRI Middleware Co., Ltd.

Issue Date : 2021/03/15

Rev : 1.06

Thank you for choosing the Aeropoint GUI Kit (Hereafter referred to as "Aeropoint kit"). This manual is a document that provides detailed instructions about how to use this product, description of the system and sample program, and a list of API's.

Before using this Aeropoint kit, thoroughly read this manual and the precautions to handle the product properly.

■ Terms of use

- This Aeropoint kit is distributed for solely evaluation of Aeropoint GUI.
  You cannot use this kit for purposes other than evaluation.
  For product development and manufacturing, please enter into a commercial agreement with CRI middleware

■ Caution

- Any unauthorized reproduction (partial or all) of this document is prohibited.
- The contents of this document are subject to change without any prior notice.
- CRI Middleware shall not be liable for any losses or damages that caused by the contents described in this document for every purpose.

■ Trademarks and Registered Trademarks

- "CRI" "Aeropoint" "CRIWARE" logo are the trademarks or registered trademarks of CRI Middleware Co., Ltd. in Japan and other countries.
- "OPTPiX" is a registered trademark of Web Technology Corp.
- "RX" "RZ/A" "R-Car" are the trademark or registered trademarks of Renesas Electronics Corporation in Japan, the United States, and other countries.
- "Microsoft Windows®" and "Microsoft PowerPoint®" are the trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.
- "Adobe Photoshop" and "Adobe PDF" are either registered trademarks or trademarks of Adobe Inc. in the United States and/or other countries.
- All other company names and product names in this document are the trademarks or registered trademarks of their respective companies.

■ Renesas Alliance Partners

- CRI Middleware Co., Ltd. is an alliance partner of Renesas Electronics Co., Ltd.

Index

1 Introduction 5
1.1 Features 5
1.1.1 Feature 1 "Creating GUI with PowerPoint®" 5
1.1.2 Feature 2 "Sound / Movie playback" 6
1.1.3 Feature 3 "High Image quality and screen stability" 7
1.1.4 Feature 4 Multilingual 8
1.2 Development by Aeropoint 9
1.2.1 Simple, fast 9
1.2.2 Operates as an independent module 10
1.2.3 Runtime SDK 11
1.2.4 For commercialization 12
2 Introduction of runtime kit 13
2.1 Reference kit 13
2.2 FIT kit 13
3 Reference kit 14
3.1 Required equipment 14
3.2 Setup 15
3.2.1 Contents of Reference kit 15
3.2.2 Aeropoint toolkit 16
3.2.3 Actual device 16
3.2.4 Connect to power supply 17
3.2.5 microSD card 17
3.2.6 Aeropoint player 18
3.2.7 LAN connection 19
3.2.8 D-Amp board connection 20
3.2.9 All devices 21
3.3 Operation check of Reference kit 22
3.3.1 Display Welcome screen 22
3.3.2 Play sound 26
3.3.3 Capture 26
3.4 Sample data 27
3.4.1 Meeting room demonstration 27
3.4.2 "Toilet seat touch panel controller" "Blood Pressure/Pulse Meter" "Home control" demo 29
3.4.3 "Sound Test" demo 30
3.4.4 "Video Test" demo 31
3.4.5 "I/O Test" demo 31
4 FIT kit 32
4.1 Supported microcontrollers 32
4.2 Preparing for e²studio and QE for Display 32
4.3 Set up the FIT kit 33
4.3.1 Create a new project 33
4.3.2 Setting up "QE for Display" 36
4.4 Create GUI 44
4.5 Library 45
4.6 Create and Execute Program 46
5 Aeropoint toolkit features 48
5.1 "Target" group 48
5.1.1 "Write All" 48
5.1.2 "Write" 49
5.1.3 "Play" 49
5.2 "Resource" group 50
5.2.1 "Text" 50
5.2.2 "Movie" 51
5.2.3 "Sound" 52
5.3 "Slide" group 53
5.3.1 Slide Name 53
5.4 "Shape" group 53
5.4.1 Shape Name 53

5.4.2 ID 53
5.4.3 Button ID 54
5.4.4 X / Y / Width / Height / Lock aspect ratio 54
5.4.5 Hidden 54
5.4.6 QR 54
5.4.7 Button 55
5.4.8 Reset 55
5.4.9 Comment 55
5.5 "Edit" group 56
5.5.1 Merge Shapes 56
5.5.2 Free form & Subtract 57
5.5.3 Copy & Apply Object Style 57
5.5.4 Bring to Front / Send to Back 58
5.5.5 Group / Ungroup 58
5.5.6 Align 59
5.5.7 Rotate 59
5.5.8 Selection Pane 60
5.5.9 Format Object 61
5.6 "Option" group 62
5.6.1 Import 62
5.6.2 Export All / Export 63
5.6.3 Create Font 64
5.6.4 Resolution 64
5.6.5 Settings 64
6 Try it 65
6.1 Preparation 65
6.1.1 Create new 65
6.1.2 Change resolution 66
6.2 Design your own GUI 67
6.2.1 Create an overall picture 67
6.2.2 Create an overall image 68
6.3 Adjustment 73
6.3.1 Place the base parts 73
6.3.2 Making your original buttons for GUI 74
6.3.3 Create by copy and paste 79
6.4 Control settings 83
6.4.1 Buttons 83
6.4.2 Switching screens 84
6.4.3 Number 86
6.5 Commercialize your prototype 88
6.5.1 Fine tuning 88
6.5.2 Check by someone else 88
6.5.3 Commercialization 88
7 SDK 89
7.1 Application development environment 89
7.1.1 "Renesas Flash Programmer" 89
7.1.2 Aeropoint Toolkit 90
7.1.3 RX72N Envision Kit 90
7.1.4 e²studio 91
7.1.5 "e² studio" Toolchain 92
7.2 System configuration 93
7.3 Basic flow 94
7.4 Aeropoint runtime 95
7.4.1 Minimal application 96
7.4.2 Display the page 102
7.4.3 Callback 104
7.5 Aeropoint programming reference 107
7.5.1 Symbol, structure, enum 107
7.5.2 UART Command 108
7.5.3 Player API 109
7.5.4 Page switching API 116
7.5.5 Video API 117
7.5.6 Sound API 119

7.5.7 The character display API 121
7.5.8 Display switching API 124
7.5.9 Test IO API 125
7.5.10 Debug Text API 128
7.5.11 Primitive display API 134
7.5.12 Data Flash API 137
7.5.13 Serial communicaiton API 139
7.6 Sample program 151
7.6.1 Switch slides 151
7.6.2 Display characters on the screen 152
7.6.3 Sound playback 153
7.6.4 Video playback 154
7.6.5 Switch visible / invisible 155
7.6.6 Control test IO 156
7.6.7 Display debug text on screen 157
7.6.8 Draw a primitive shape on the screen 159
7.6.9 Serial communication 160
8 How to connect to other equipment 163
9 Make Font 164
9.1 "FontMaker" tool launch 164
9.2 How to use 165
9.2.1 Pull-down menu 165
9.2.2 "Size of 1 character block" 165
9.2.3 "Font size" 165
9.2.4 Preview 165
9.2.5 Position correction 165
9.2.6 Save 165
10 Program writing 166
10.1 Renesas Flash Programmer 166
10.2 Writing procedure 166
10.2.1 Change to write mode 166
10.2.2 Write the program with Renesas Flash Programmer 167
10.2.3 Change to single boot mode 168
11 Write data image 169
11.1 Things to prepare 169
11.2 Writing procedure 169
12 Q&A 171
12.1 Aeropoint toolkit add-in does not appear 171
12.2 When using slide theme or master slide, shapes are not placed correctly on the actual device 173
13 Reference price 174
14 Contact us 174
14.1 CRI Middleware 174
15 Revision record 174

1 Introduction

Welcome to "Aeropoint GUI"
"Aeropoint GUI" is a graphical UI middleware that provides rich visual representation on microcontrollers in a reasonable price range.
Hereafter, we refer "Aeropoint GUI" as "Aeropoint".

1.1 Features

There are four features of Aeropoint.

1.1.1 Feature 1 "Creating GUI with PowerPoint®"

In the field of embedded microcontrollers, more and more devices are equipped with color LCD and touch-able GUI (Graphical User Interface). Nowadays users are accustomed to smartphones that are with wide color display, high resolution and touch screen operation, there is an increase in demands for pursuing such "display" or "sense of operation" during developments.

You can easily add-in Aeropoint in PowerPoint® to create your GUI's that can reduce your production cost.

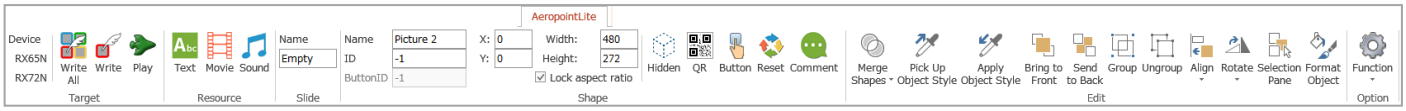


Fig. 1 Aeropoint Add-in

In addition, CRI takes the advantage of the know-how technology of the Game industry, our "Aeropoint Runtime Engine" helps to easily create almost all the display control (GUI) that satisfies the "sense of operation".

Fig. 2 Sample GUIs created by PowerPoint®

1.1.2 Feature 2 "Sound / Movie playback"

By using CRI's "D-Amp Driver", you can simply set ON / OFF signal of microcontroller to experience the high quality sound playback.

【 Sound playback 】

- Use the circuit digitization to reduce cost for the analog devices.
- No need to worry about the risk of discontinuation of LSI and/or other analog devices needed, making long-term use and long-term supply guarantee possible.
- Other than the microcontroller, only transistors and resistors are required for hardware other than the microcomputer. Thus suitable for devices use in harsh operating environments.
- Able to play two monaural sounds (24KHz) simultaneously.

Fig. 3 D-Amp Driver

By utilizing CRI's advanced streaming technology from the game field, we are the first to offer reasonable prices for video playbacks in microcontroller. You can use GUI contents or incorporating video advertisements.

【 Movie playback 】

- The only kit that can play video in RX65N / RX72N (as of 2020/06/25) with sound playback.
- Can play up to 8 videos of 15fps and 30 seconds. Can add more videos if attach to an external memory.
- Able to encode any videos including uploaded videos or commercials.

Fig. 4 Example of Video played on the device

1.1.3 Feature 3 "High Image quality and screen stability"

1. High image quality

The built-in "OPTPiX", which is famous for its excellent color reduction technology, can display high resolution pictures in GUI by limiting to 256 colors.

Aeropoint uses a memory-saving and two-screen display mechanism to achieve both high image quality and stability.

You can reduce the full-color image to 256 colors with OPTPiX up to 1/4 of its size.

Simple color reduction

Color reduction (256 colors) by OPTPiX

Fig. 5 Difference in image quality depending on the color reduction method

2. Screen stability

The reason why the screen flickers is due to displayed screen updates. Therefore, CRI prepares two screens internally to take advantage of the unused memory. By reading the image on the "hidden screen" and switching the screen at the moment when the liquid crystal display stops drawing, which can help to increase the screen stability. The same method also applies to video playbacks.

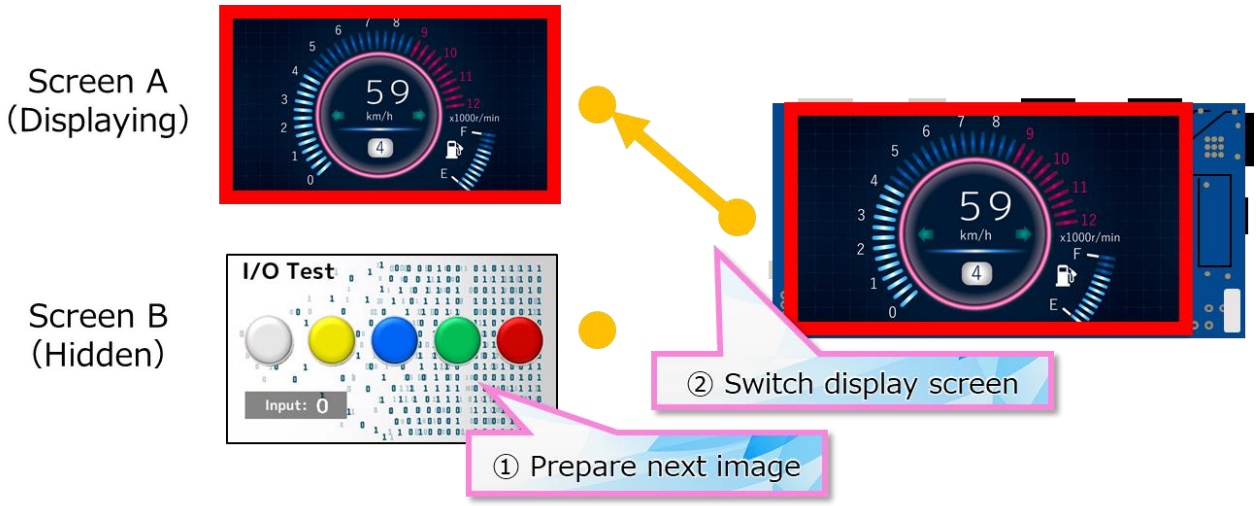


Fig. 6 Update display by using the "displayed" and "hidden" screens

1.1.4 Feature 4 Multilingual

You can shift languages to save the development time. 8 kinds of fonts are provided in Aeropoint.

Fig. 7 Multilingual

You can use the fonts in your PC or create fonts for the actual device by our tools.
You can create any fonts you wish to fit your product or screen image.

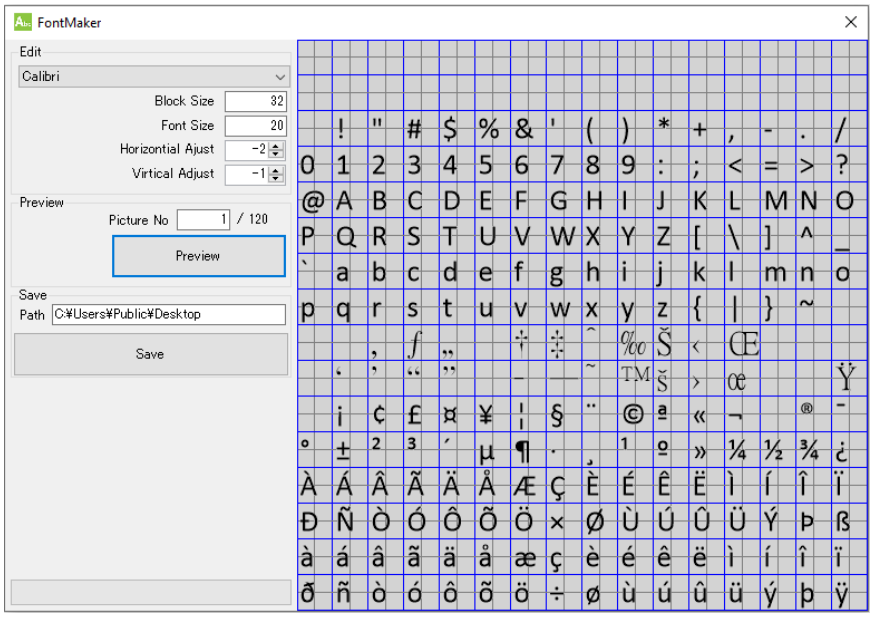


Fig. 8 Create font tool

Note: You have to contact the font IP companies for the font usage rights.
All fonts registered as standard on PC are owned by the IP companies that own the fonts, and cannot be reused without their permission.

1.2 Development by Aeropoint

1.2.1 Simple, fast

Aeropoint is very easy to use. You can design with PowerPoint®, set the GUI behavior with the Aeropoint add-in. Click "Write" to transfer to the actual device to prepare for playback.

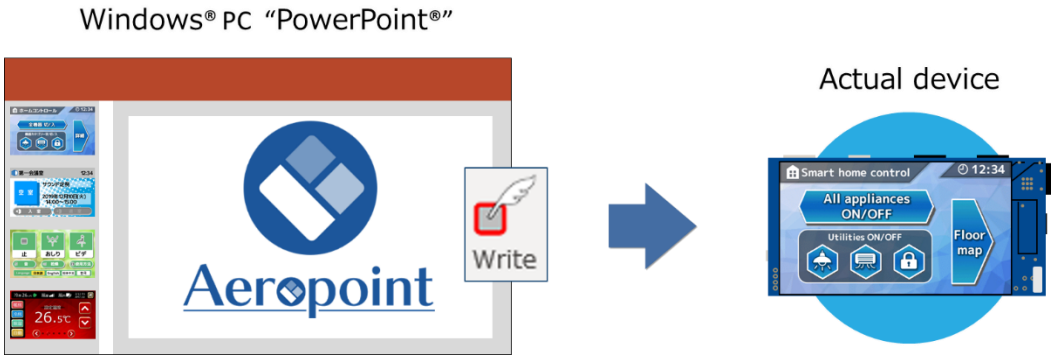


Fig. 9 Simple configuration

It takes a few seconds to transfer and display the data on the actual device. And you can start anytime. There is no need to create any programs or materials using dedicated tools, so you can more time for GUI designs.
Another advantage is that you can share the information for design updates or localization for multilingual support in one single. PowerPoint® file among GUI designers, translators, and other in-charges

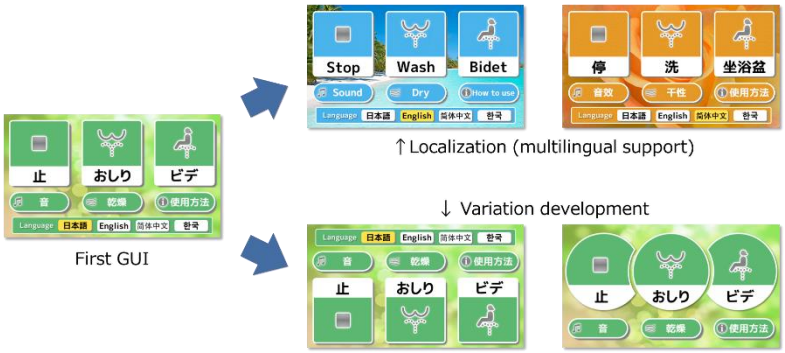


Fig. 10 Easy to change languages by using the original

Note that even if the customer's product uses the same microcontroller as the sample board, customization is required due to differences in peripheral devices and architecture. For commercial users, you can contact CRI for the customization to be done in CRI (Refer to 14 Contact us)

.

1.2.2 Operates as an independent module

The GUI created by Aeropoint works as an independent module, so you can attach it externally without affecting the major design of the product. The existing MPU uses commands to control the GUI to display its status.

Fig. 11 Control and measurement using commands

The following commands are some of the examples you can use.

Control	Measurement
Slide display switching	Touch button
Shape display switching	Log
Text display	Screen capture
Sound playback	Status
Video playback	Data

Fig. 12 Command example

In this Aeropoint kit, connect the "Connector" tool that runs on the PC, with wired LAN instead of the MPU to communicate with the actual device and measures basic control and logs.

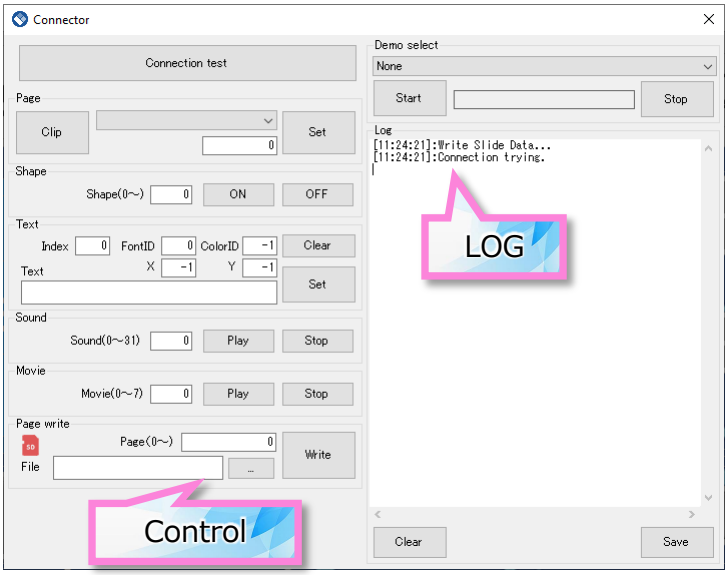


Fig. 13 "Connector" on PC

1.2.3 Runtime SDK

You can also run C language in Aeropoint's runtime SDK, which is easy to install. For example, use Aeropoint functions by main.c as below.

```
void main(void)
{
    CriBool flag = CRI_TRUE;

    /* Initialize */
    CriAeroPlayer_Initialize();

    /* Main loop */
    while( flag )
    {
        /* Main */
        flag = CriAeroPlayer_Main();
    }

    /* Finalize */
    CriAeroPlayer_Finalize();

    return;
}
```

Fig. 14 The most compact program

On the other hand, to use the GUI, a memory capacity corresponding to the resolution is required. Also, you need resources for the CPU and peripheral devices.

Renesas integrated environment "e² studio" is used for the programming of the runtime SDK.

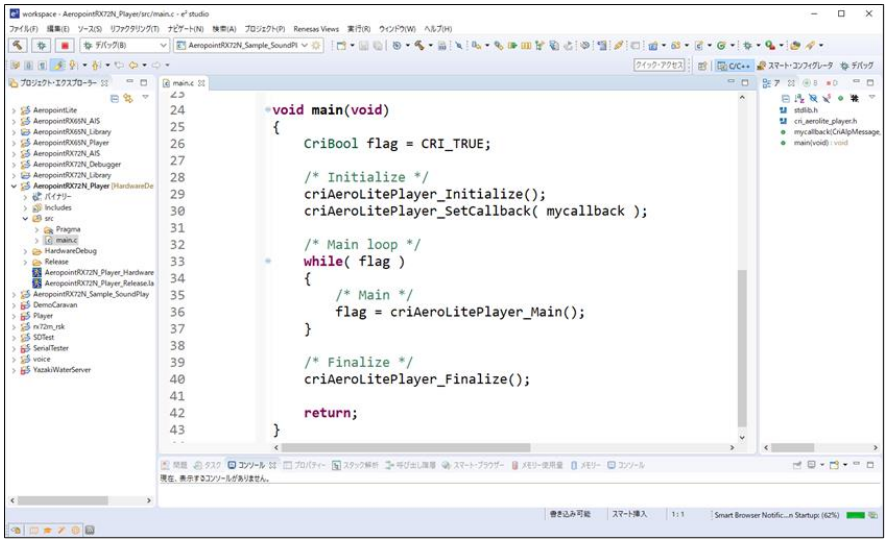


Fig. 15 e²studio

1.2.4 For commercialization

After the verification using this Aeropoint kit is completed, you will be ready for trial production or product development.

If you want to develop on your board (or any other board), CRI needs to fit Aeropoint library to your hardware. Please contact us by referring to "14 Contact us".

2 Introduction of runtime kit

Aeropoint's runtime kits are available in two types, "Reference kit" and "FIT kit", according to the customer's purpose.

2.1 Reference kit

"Reference kit" is perfect for those who want to try Aeropoint first. With the RX72N Envision Kit from Renesas Electronics, you can experience all the features of Aeropoint. There are many UI samples, and you can display the UI created in PowerPoint® and control it by programming. For details, see "3 Reference kit." in this manual.

Fig. 16 RX72N Envision Kit

2.2 FIT kit

"FIT kit" is used by those who want to proceed with product development and mass production after trial use with Reference kit. All Aeropoint functions are provided as FIT modules in order to realize operation on the user board. Using the QE for Display tool provided by Renesas Electronics, it is possible to install Aeropoint on the user board in a wizard format. For details, see "4 FIT kit." in this manual.

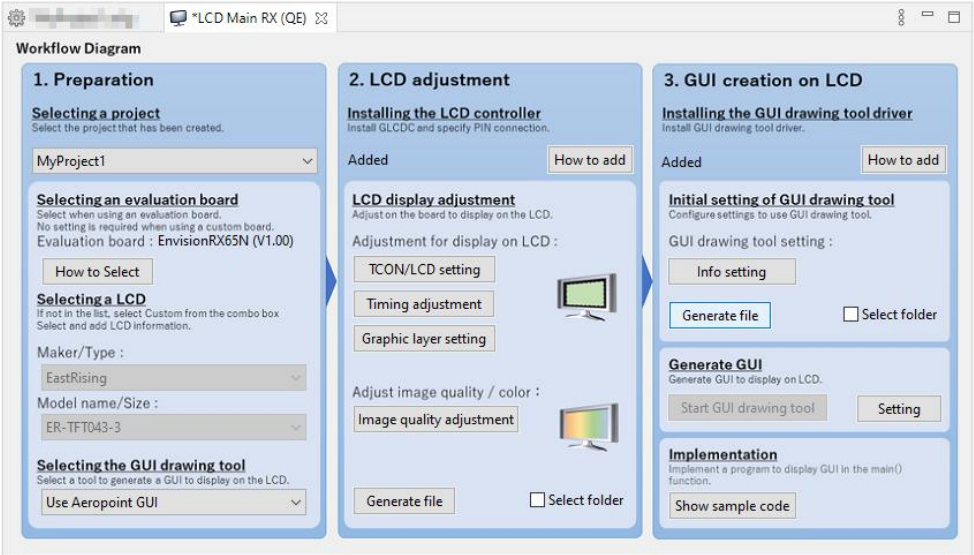


Fig. 17 Introduction using QE for Display

3 Reference kit

Let's experience Aeropoint

3.1 Required equipment

Check the required equipment and software.

- Reference kit
  A simple evaluation package for the purpose of experiencing Aeropoint. Includes the following.

  - ✧ Basic Software
    - AeropointReferencekit_xxxxxxxx.zip
      ※ "xxxxxxxx" is release date.
  - ✧ Actual device
    - RX72N Envision Kit
  - ✧ Peripheral device
    - USB male ←→USB Type-B male cable
      ※ This cable plugs into a USB port and is used for power supply and development using e² studio
    - microSD card
    - D-Amp board, connection cable, speaker
      ※ Used for playing sound.

- PowerPoint®
  PowerPoint® that can be used is a stand-alone version of 2013 or later or Office 365 version. Web version is not available.

- USB-LAN adapter, LAN cable
  It is used for exchanging data, etc. by connecting the PC to the actual device. If your PC has a free LAN port, you don't need a USB-LAN adapter.

3.2 Setup

Once you have the equipment and software, set up.

3.2.1 Contents of Reference kit

Extract the following packages.

- Reference kit
  - AeropointReferencekit_xxxxxxxx.zip

    ※ "xxxxxxxx" is release date.

Check the structure of folders and files.

```
AeropointReferenceKit_xxxxxxxx
│   AeropointUsersManual_xxxxxxxx_jp.pdf ── This document
│
├──Toolkit
│       AeropointLiteToolkit_64bit.exe   ── Aeropoint's toolkit. Install it on your PC
│                                            and add the add-in to PowerPoint®.
│       AeropointLiteToolkit_32bit.exe   ── Please select the one that matches the OS
│                                            of your PC (64 / 32bit)
│
├──Runtime
│       AeropointRX72N_Library.zip ──────── Aeropoint runtime library that runs on RX72N.
│
│       AeropointRX72N_Player.zip  ──────── The player project used in the user application.
│
└──Samples
    ├──Player
    │       Player.mot ──────────────────── The execution program of the Aeropoint debugger.
    │
    ├──SDImage
    │       SDImage_jp.bin ──────────────── Data image of SD card.
    │
    ├──Project
    │       *.zip ───────────────────────── The Sample\Project\ to build and use with e² studio.
    │                                        You import it to e² studio and use it.
    └──GUI
            *.pptx ──────────────────────── PowerPoint® file containing GUI data
```

Fig. 18 Reference kit folder and file structure

3.2.2 Aeropoint toolkit

Install Aeropoint toolkit on your PC.

1. Quit PowerPoint® and other office products if they are running.
2. Execute the installer and follow the instructions to install.
   → 64-bit version : AeropointLiteToolkit_ 64bit.exe
   → 32-bit version : AeropointLiteToolkit_ 32bit.exe
3. After successful installation, launch PowerPoint®.

It is successful if the following ribbon menu is added to PowerPoint®.

Fig. 19 Aeropoint Add-in ribbon menu

3.2.3 Actual device

Unpack the RX72N Envision Kit and take out the main unit.

- RX72N Envision Kit
  - https://www.renesas.com/products/software-tools/boards-and-kits/eval-kits/rx72n-envision-kit.html

図 20 RX72N Envision Kit

3.2.4 Connect to power supply

To power the Envision Kit, connect the ECN1 connector to the power source with a USB cable. You can also use the USB port of the PC as the power source. For development using e² studio, connection with a PC using this cable is essential.

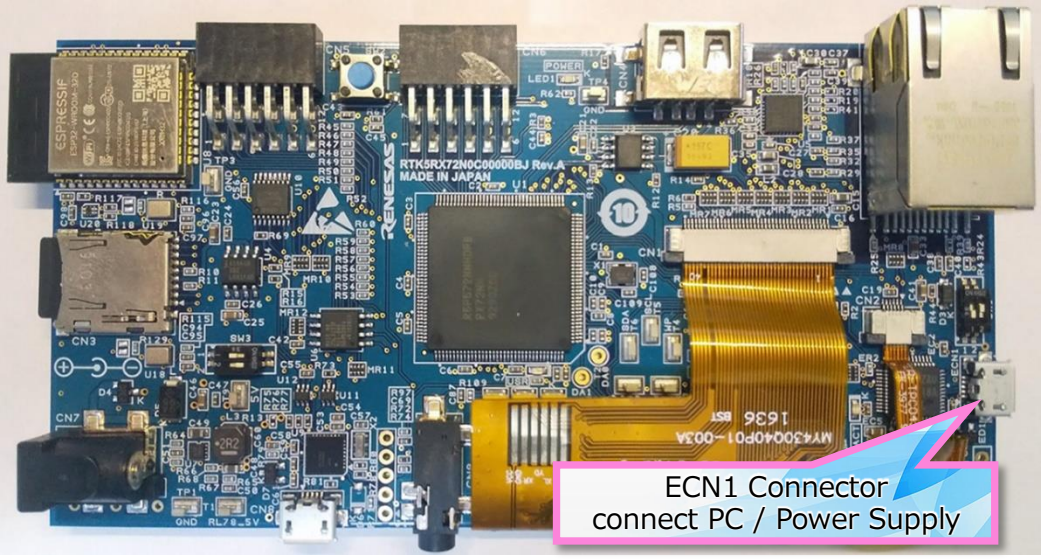


Fig. 21 Connection port for power supply

3.2.5 microSD card

The SD card that comes with Reference kit contains the data necessary to experience Aeropoint in advance. Insert the SD card into the SD card slot of the actual device.

When preparing the SD card by yourself, write "SDImage_xx.bin" included in Reference kit by referring to "11 Write data image" in this manual.

3.2.6 Aeropoint player

Aeropoint debugger for operating the GUI in cooperation with the Aeropoint toolkit is pre-written on the actual device included in Reference kit. In order to automatically start the Aeropoint debugger, set the dip switches on Envision Kit to the positions shown below.

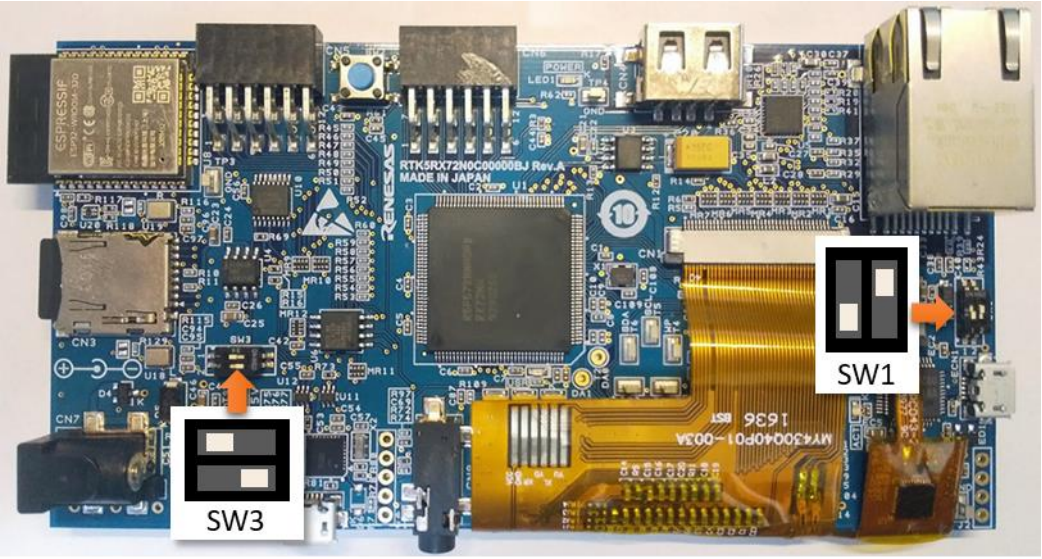


Fig. 22 DIP switch settings

To return to the Aeropoint debugger after executing the sample programs, write "Player.mot" attached to the Reference kit by referring to "10 Program writing" in this manual.

3.2.7 LAN connection

A LAN connection is used to exchange data between the Aeropoint toolkit and the actual device. If the LAN port of the PC is available, connect it to the actual device with a LAN cable. If there is no LAN port, prepare a USB-LAN adapter and connect.

Fig. 23 USB-LAN adapter

Set the "Properties" of the adapter by selecting IPv4 and change the "IP address" and "Subnet mask".

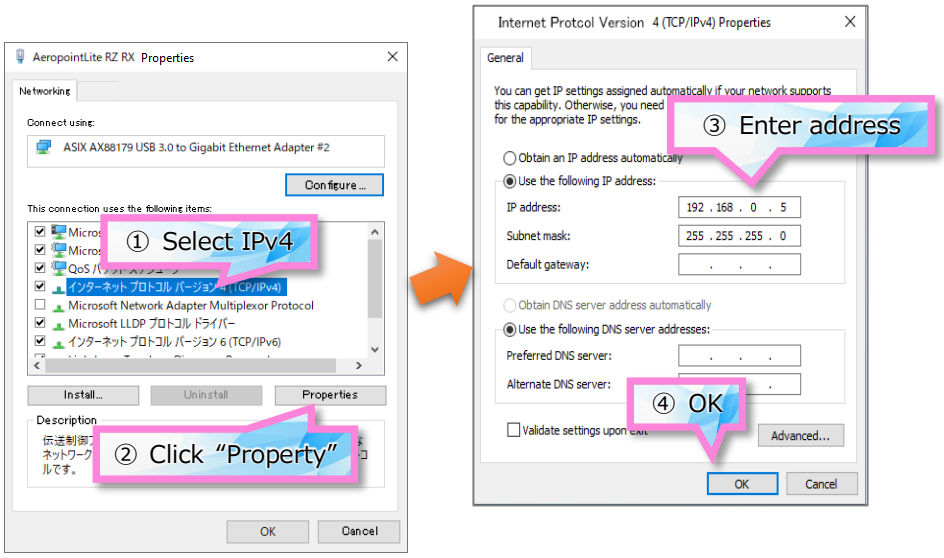


Fig. 24 USB LAN adapter - "Properties"

IP address : 192.168.0.5

Subnet mask : 255.255.255.0

The IP address of the actual device is fixed at 192.168.0.2 by the program of the runtime.

The Aeropoint toolkit accesses the "192.168.0.2" address, which is connected to the LAN adapter with the "192.168.0.5" address, as the actual device. The port uses 7743.

3.2.8 D-Amp board connection

Connect the D-Amp board and the speaker to play the sound with the "D-Amp Driver.

There are two types of D-Amp boards, so check the connection method while comparing the board you have with the figure below. Align the surface of the Common D-Amp board (the one with the speaker terminal) with the LCD surface, and connect it to the CN6 terminal (PMOD) near the center of the front using the included connection cable. The RX72N dedicated D-Amp board is directly plugged into the CN6 terminal.

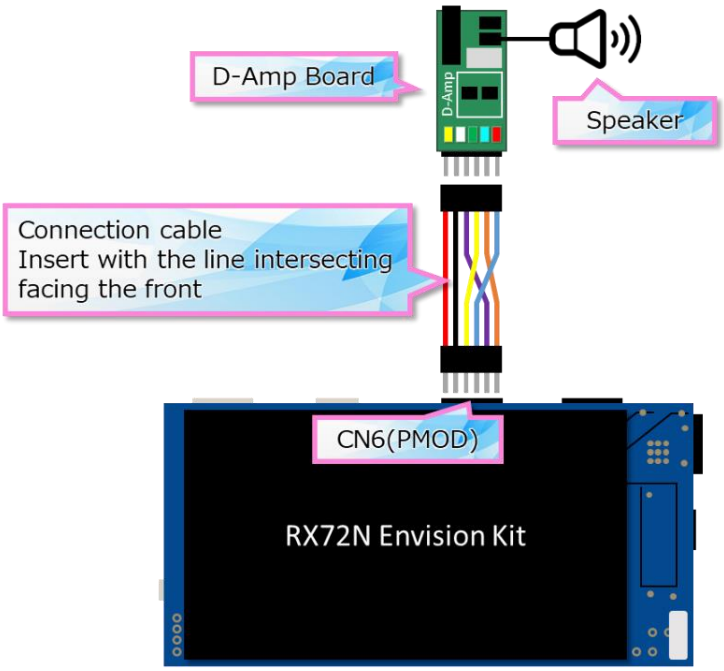


Fig. 25 Common D-Amp board connection

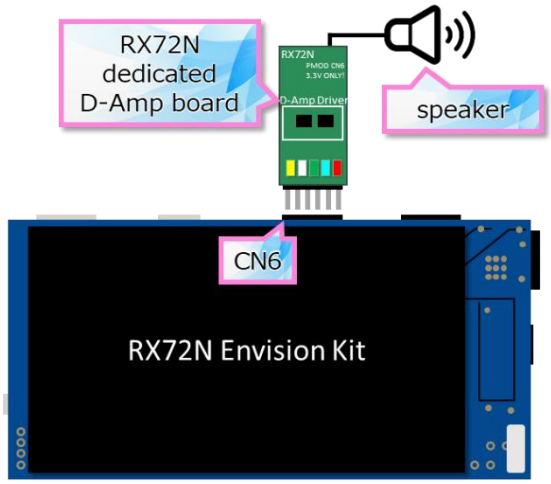


Fig. 26 The RX72N dedicated D-Amp board connection

After connecting the board, connect the speaker.
Please connect to the favorite terminals of "J2", "J3" and "J4" on the common D-Amp board.
For the RX72N dedicated D-Amp board, connect it to the desired terminals of "J1" and "J2" on the back side.
There are two speaker wires, but you can connect either pin to each terminal.

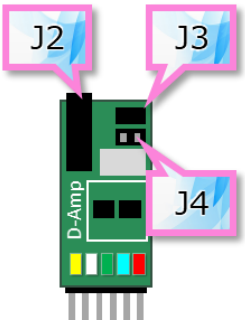


Fig. 28 Common D-Amp board speaker connection

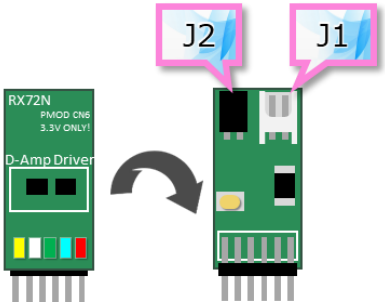


Fig. 27 The RX72N dedicated D-Amp board speaker connection

3.2.9 All devices

It will be as follows when all devices are connected.

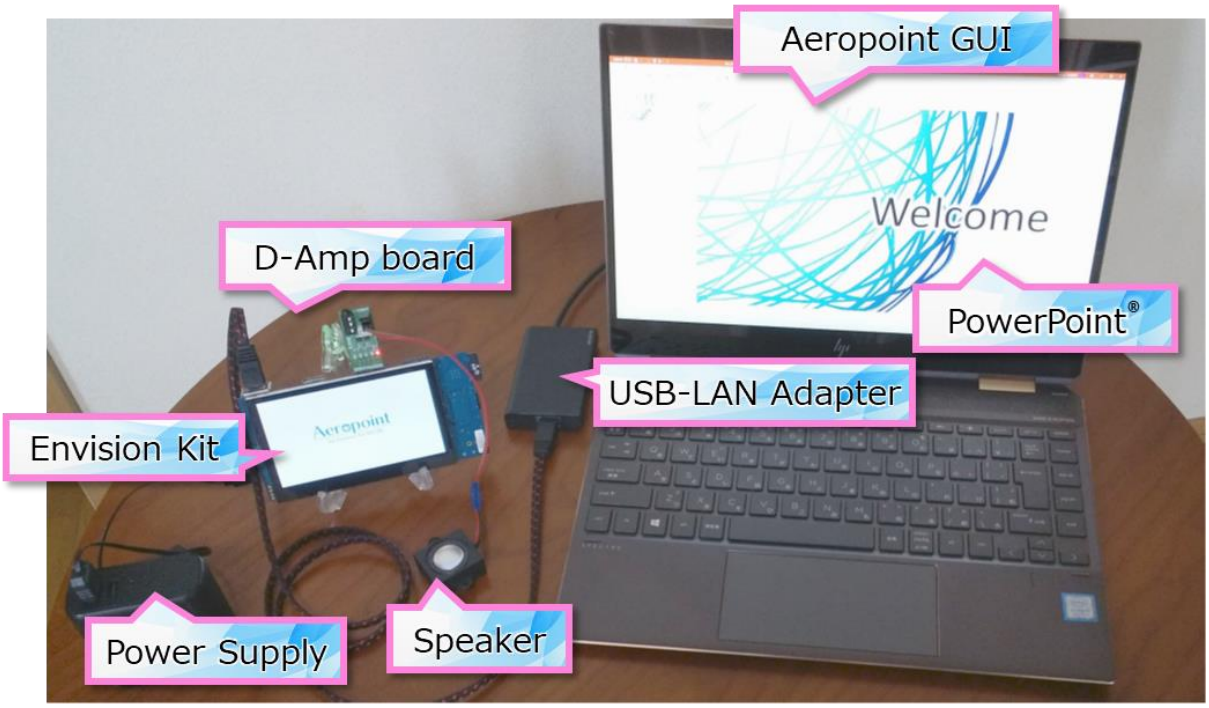


Fig. 29 Example of how to connect to all devices

3.3 Operation check of Reference kit

3.3.1 Display Welcome screen

We will use the Welcome sample to check if the connected equipment works properly. When you turn on the RX72N Envision Kit, the startup / standby screen is displayed.

Fig. 30 The startup/standby screen

Then press "SW2".

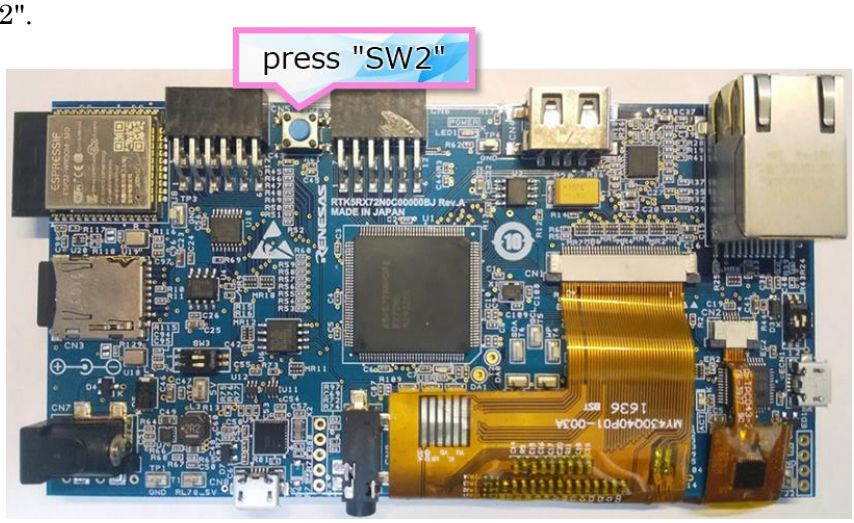


Fig. 31 press "SW2"

Aeropoint Debugger reads and displays the Welcome page registered in the SD card.

Fig. 32 Welcome screen

Press "SW2" again to return to the startup screen.

Next, start PowerPoint® on your PC and open "Sample\GUI\Sample_Welcome.pptx" included in Reference kit.

Select the "Aeropoint Lite" tab at the top of PowerPoint® and click the "Play" button displayed on the ribbon.

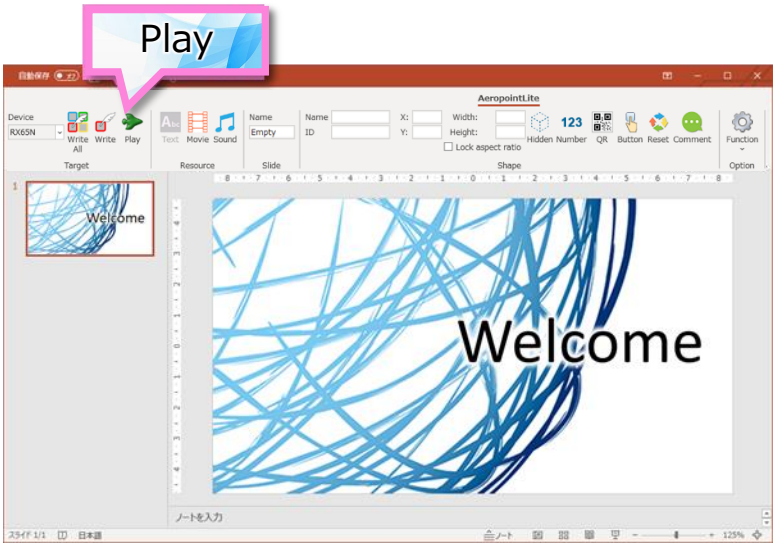


Fig. 33 Play Welcome sample

The "Connector" application, which acts as an intermediary between the Aeropoint toolkit and the actual device, starts and issues a command to display the first slide.

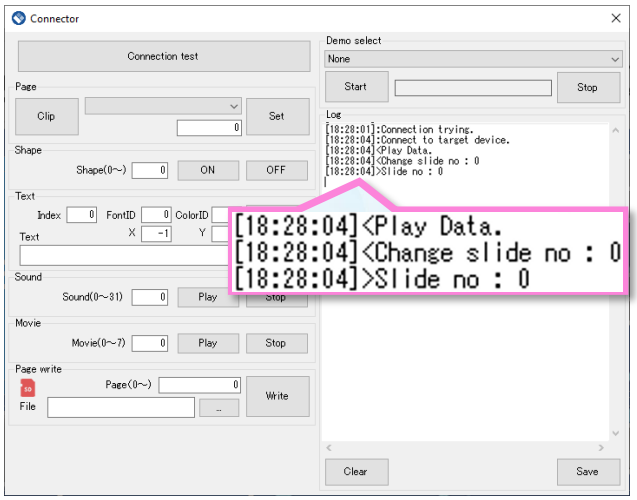


Fig. 34 Command on "Connector"

When the command arrives at the actual device, the Welcome page will be displayed as before.

Fig. 35 Welcome screen is displayed again

If the error is not displayed properly and an error is displayed in "Connector" log, a timeout may have occurred.
Try closing the "Connector" or turning the power of the actual device off and on again to see if the connection is successful. If that does not work, review the settings in "3.2.7 LAN connection" and check the IP address and so on.

Press the play button to display the first slide stored in the SD card. To display the page created by PowerPoint®, you need to update the contents of the SD card.

I added an appropriate shape to the first Welcome slide. (see next Fig.)

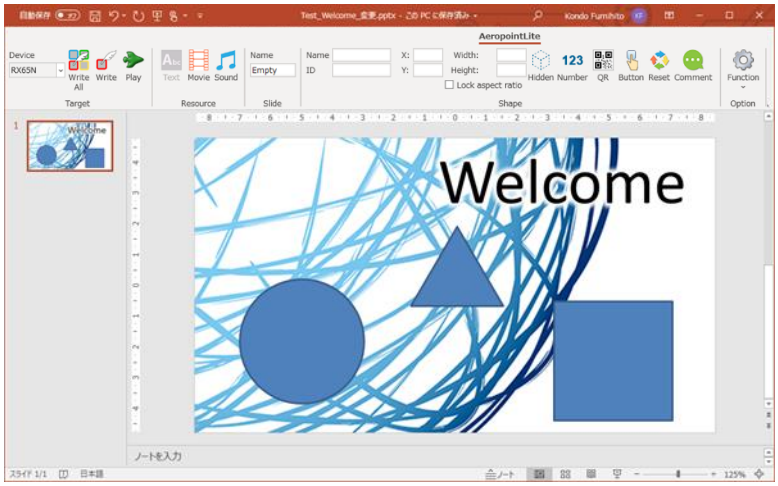


Fig. 36 Added an appropriate shape to the first Welcome page.

Then click "Write All". When all the data has been written to the actual device, a message indicating the end of writing will be displayed in "Connector" log.

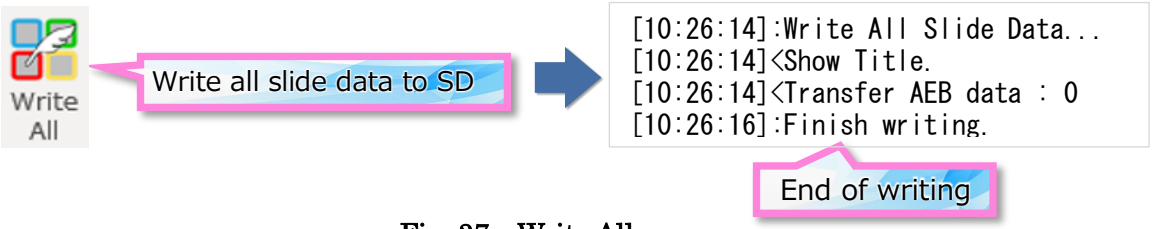


Fig. 37 Write All

If you click "Play" again and the content displayed on the actual device is updated, you are successful.

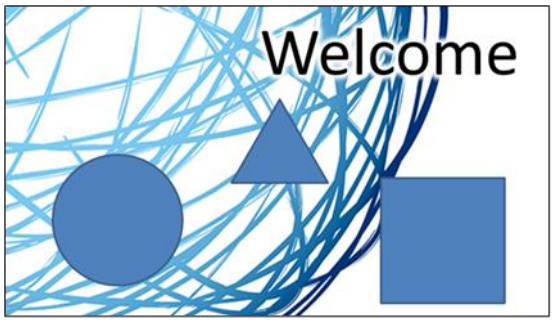


Fig. 38 The changed content is displayed

If the writing completion message is displayed immediately and the writing is not actually completed, an internal SD card error may have occurred. Please reinsert the SD card and check again.

3.3.2 Play sound

Connect the D-Amp board and the speaker correctly. After inputting an appropriate value (about 0 to 5) in the "Sound number" of the "Connector", click "Play". Sound is played from the speaker connected to the D-Amp board.

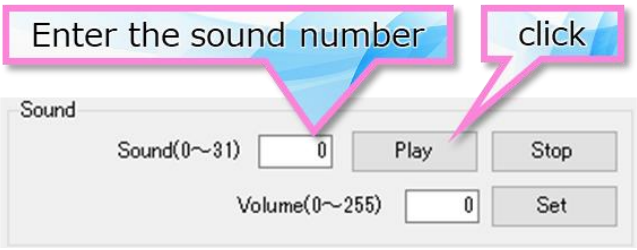


Fig. 39 Play sound

3.3.3 Capture

If you click "Clip" on the connector and wait for a while, the screen displayed on the actual device will be copied to the clipboard of the PC.

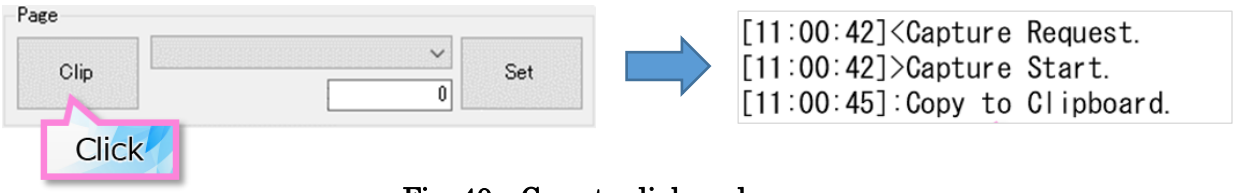


Fig. 40 Copy to clipboard

Use the captured image by pasting it in an application such as Paint.

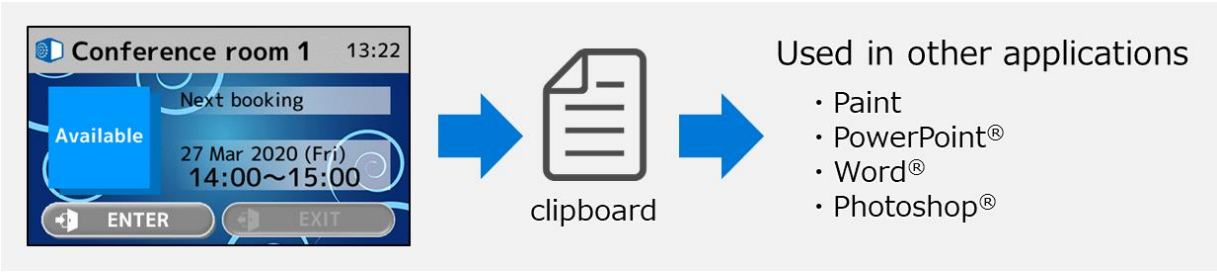


Fig. 41 Used in other applications

If these mechanisms work well, Reference kit operation check is completed.

3.4 Sample data

Reference kit has some sample data.
You can check the behavior of these samples using "Connector".

3.4.1 Meeting room demonstration

Select "Meeting Room" demo from "Connector" demo list. Click "Start".

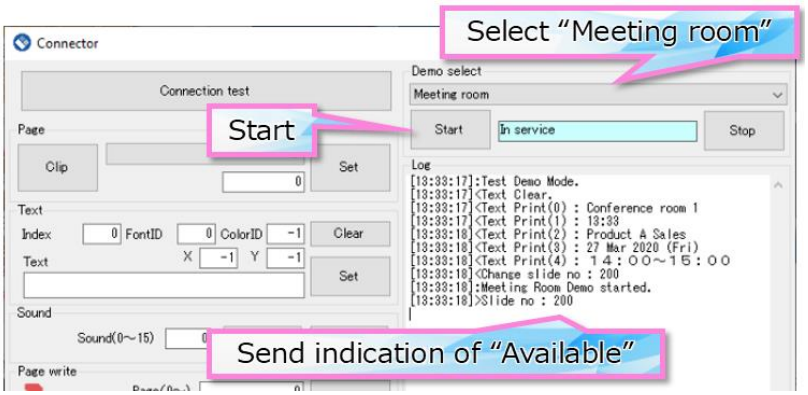


Fig. 42 Launch "Meeting Room" demo

"Connector" will start the demo program, the necessary text information, such as "Meeting room name", "Time", and "What kind of meeting", will then be sent to the "Meeting Room" demo. After that, "Connector" will instruct "Available" information data to be shown on the screen.
If the above information is displayed on screen, that means the instructions are successfully received.

In this "Meeting Room" demo, you can also switch the displays by pressing the "Enter" and "Exit" buttons.

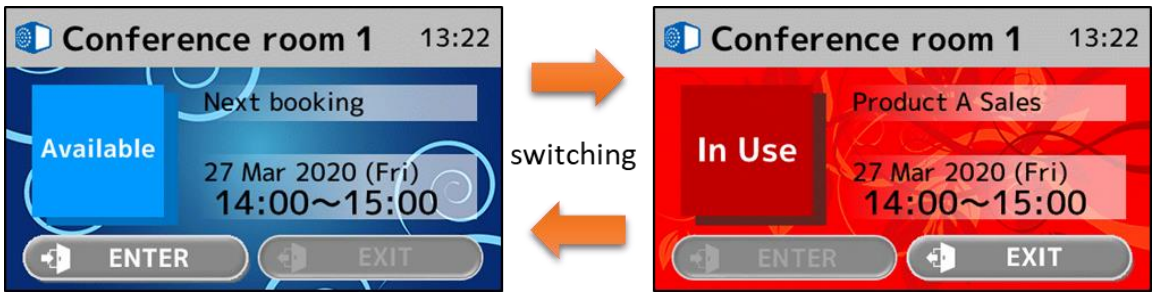


Fig. 43 "Meeting room" Demo - Screen Switching

Connector demo program can detect minute-by-minute changes in time; and will update with the current time.

```
[11:38:00]<Text Print(1) : 11:38
[11:38:00]<Update Screen.
[11:39:00]<Text Print(1) : 11:39
[11:39:00]<Update Screen.
[11:40:00]<Text Print(1) : 11:40
[11:40:00]<Update Screen.
[11:41:00]<Text Print(1) : 11:41
[11:41:00]<Update Screen.
```

Fig. 44 Display current time

The black text displayed in "Meeting room" demo is displayed using a font system that allows the content to be changed dynamically. You can also use "Connector" to change the texts.

You can change the name of the meeting room. Set the index as "0", enter your new text and click "Set".

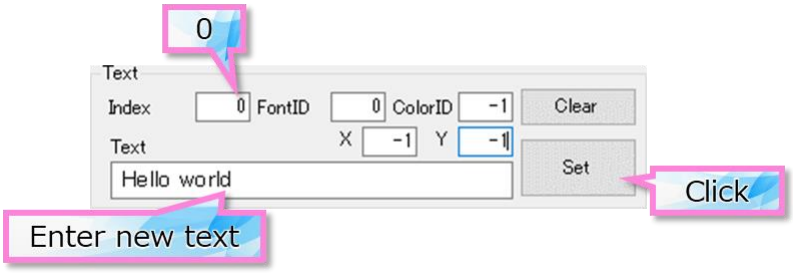


Fig. 45 Change the text from "Meeting room" to other texts

The actual device will show the new texts on the screen correspondingly.

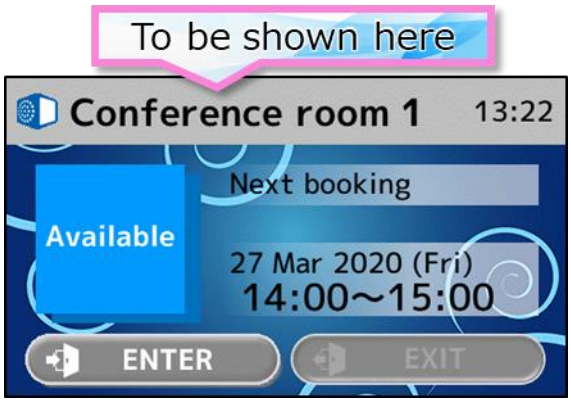


図 46 New display

In this demo, Japanese font sample is stored in Font ID "0". Character code is UTF-8. Any languages that support UTF-8 (max 8 font styles) can be used. For details, see「9 Make Font」

Press the "Stop" button at the top right of "Connector" to end the demo. It will then return to the startup/standby screen.

3.4.2 "Toilet seat touch panel controller" "Blood Pressure/Pulse Meter" "Home control" demo

There are some other demos, so let's start with the "Connector".

We created a multilingual screen of "Toilet seat touch panel controller" by copying and pasting and changing the color and background. This is a method that takes advantage of the convenience of creating screens with PowerPoint®.

Fig. 47 "Toilet seat touch panel controller" demo

"Blood Pressure/Pulse Meter" demo shows the changes in graphs and numerical values.
Random preset data runs automatically and the display refreshes every few seconds.

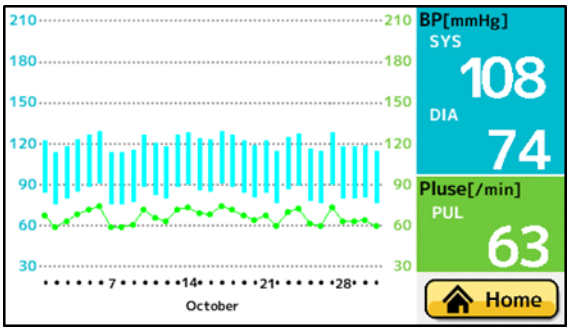


Fig. 48 "Blood Pressure/Pulse Meter" demo

"Home control" demo can control various GPIOs to ON/OFF the LEDs.

Fig. 49 "Home control" demo

3.4.3 "Sound Test" demo

Select "Sound Test" to start this demo.

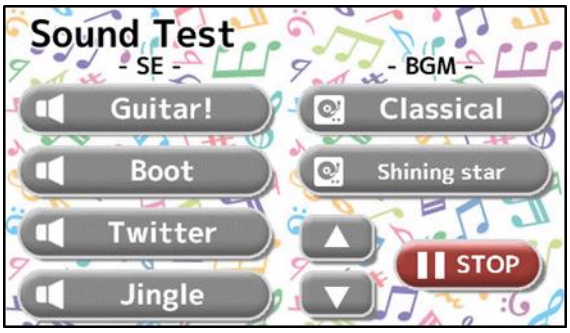


Fig. 50 "Sound Test" demo

Whenever you press a button, the demo will go through the D-Amp board and the sound will then be played through the connected speaker. The Left column is SE while the Right column is BGM.

The BGM and SE can be played separately or simultaneously.

"D-amp Driver" has implemented "streaming playback mode" from the SD card, so that you can play "long" duration sound.

Aeropoint also reads information about demo screens and fonts from the SD card. During playback, dynamic loading is performed to prevent the streaming being stopped.

3.4.4 "Video Test" demo

Select "Video Test" to start this demo.

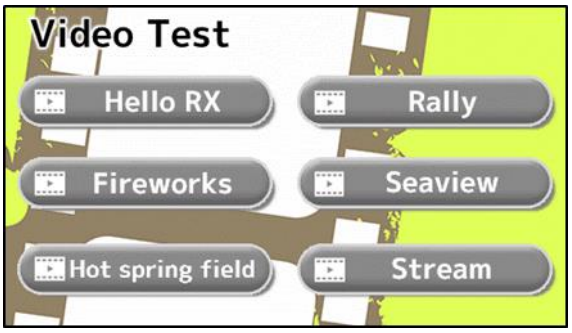


Fig. 51 "Video Test" demo

Press a button to play the corresponding video.
Touch the video playback to return to the menu page.

This demo consists of uploaded videos from public video sharing platforms and CRI's video library. Note that this demo, including the above screenshot, CANNOT be used without an NDA as some videos may need permission beforehand.

3.4.5 "I/O Test" demo

Select "I / O Test" to start this demo.

Turns on and off the LED on the D-Amp board and senses the rear switch.

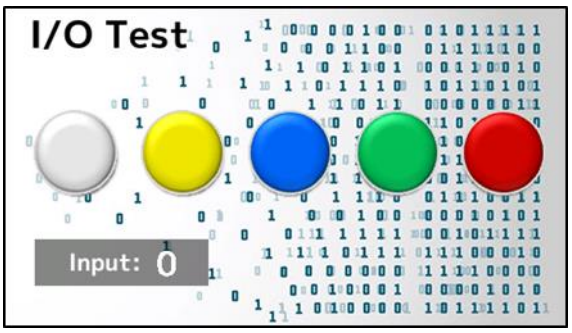


Fig. 52 "I/O Test" demo

4 FIT kit

The FIT kit runs Aeropoint on your custom board. If you don't use the FIT kit and want to know how to make it using the toolkit, skip this chapter.

4.1 Supported microcontrollers

The FIT kit currently supports the following microcontrollers

Family	Microcontrollers
RX	RX651, RX65N(ROM capacity: 1.5MB-2MB only) RX72M, RX66N, RX72N

Fig. 53 Supported microcontrollers

We are working on microcomputers that are not shown here. If you have any requests, please contact CRI.

4.2 Preparing for e²studio and QE for Display

To use the FIT kit, you need to install "QE for Display" in e2studio beforehand. Please refer to the following Renesas website to install it. If you select "QE for Display" when you install the 2021-01 version of e²studio, you can install it at the same time.

[e² studio]
https://www.renesas.com/software-tool/e-studio

[QE for Display: Development Assistance Tool for Display Applications]
https://www.renesas.com/software-tool/qe-display-development-assistance-tool-display-applications

4.3 Set up the FIT kit

In this section, we will start with a new project using the FIT kit, and explain how to display the screen on the actual device.

This time, we will use the RX65N Envision Kit from the initial state, instead of the RX72N Envision Kit used in Reference kit. We will use the RX65N Envision Kit from the initial state instead of the RX72N Envision Kit, so the flow will be similar to that of a customer using a custom board.

4.3.1 Create a new project

Start by creating a new project.
To use the RX65N Envision Kit, select "File" > "New" > "Renesas C/C++ Project" > "Renesas RX"

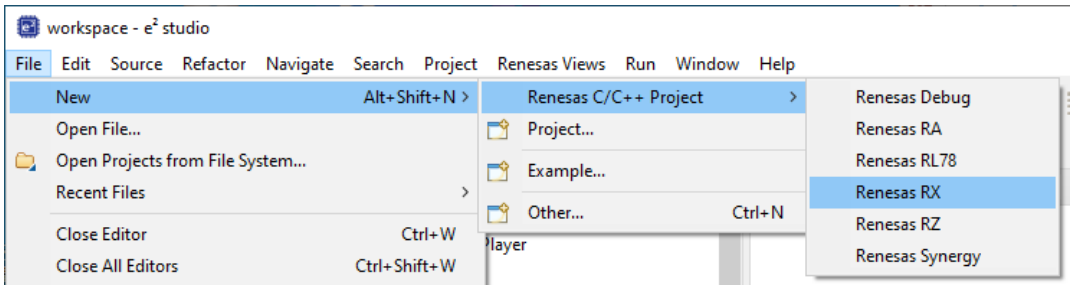


Fig. 54 Create a new project

In the "Templates for Renesas RX project" dialog, select "Renesas CC RX C/C++ Executable Project" and click "Next".

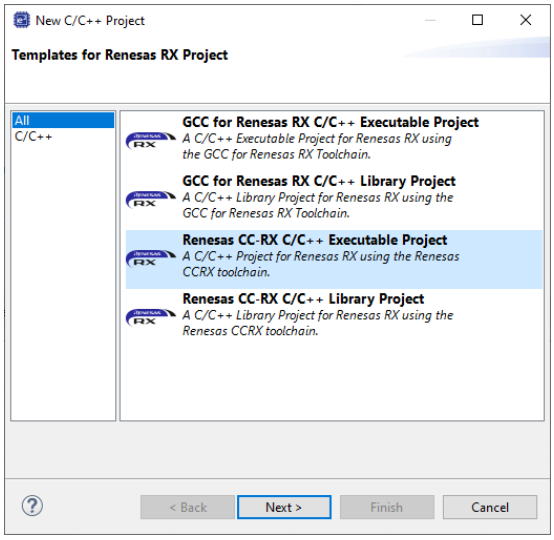


Fig. 55 Select "Executable Project"

In this example, we have chosen "MyProject" as the project name. Please use only one-byte alphanumeric characters in the project name, and do not start with a number.

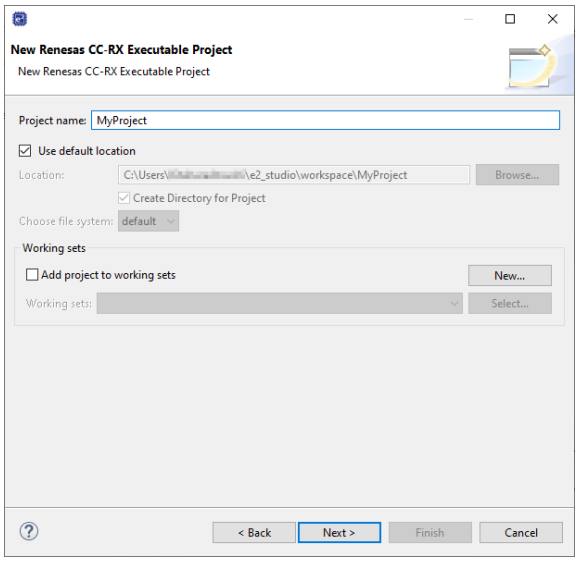


Fig. 56 Enter project name

For "Target board", select "EnvisionRX65N". For "Create Hardware Debug Configuration", select "E2 Lite"

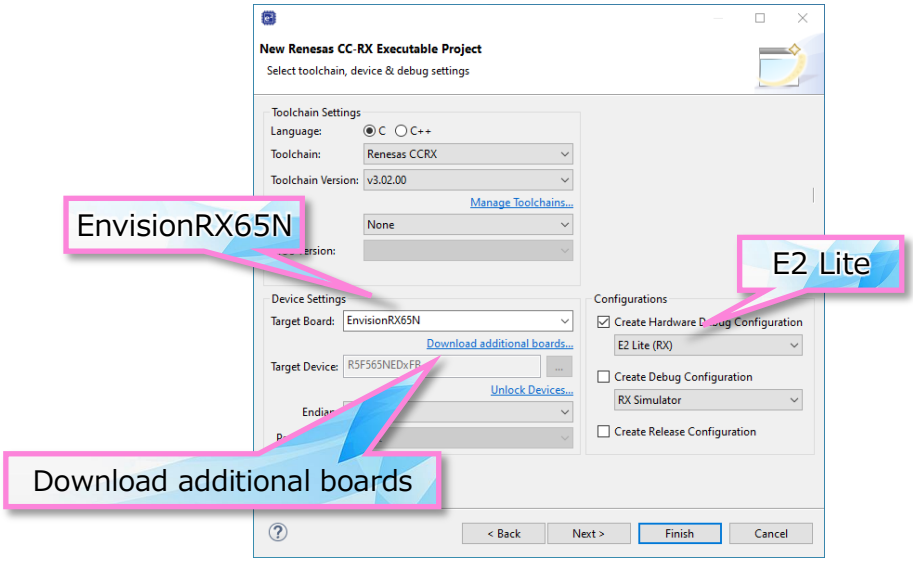


Fig. 57 Select "Target Board" etc.

If you don't see that option here, click "Download additional boards..." below the field to add the board.

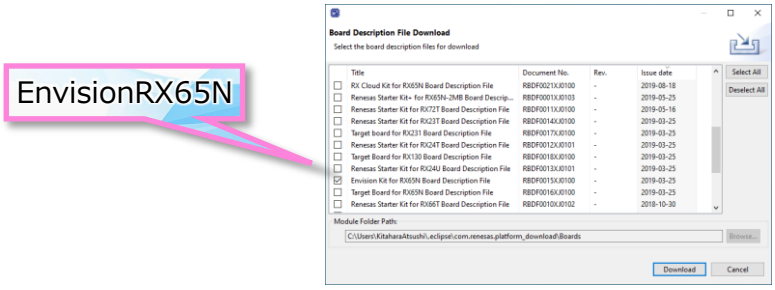


Fig. 58 Example "board select screen"

After checking the "Smart Configurator" check box, click "Finish" to generate the project.

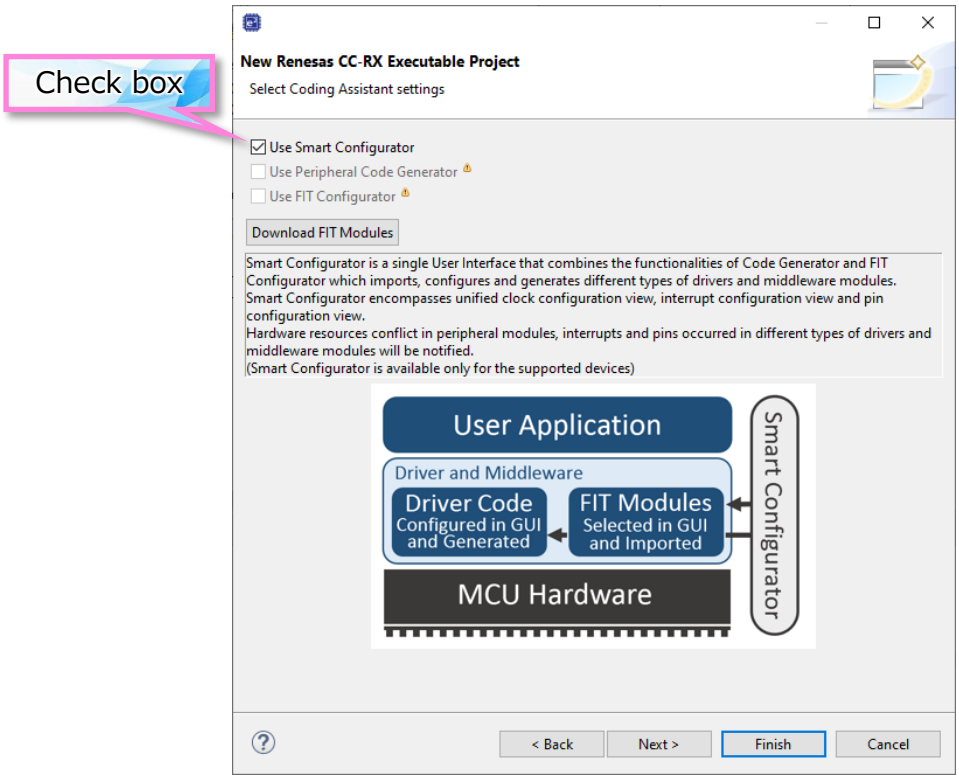


Fig. 59 Check "Use Smart Configurator"

After the project generation is complete, the configured "MyProject" project will appear.

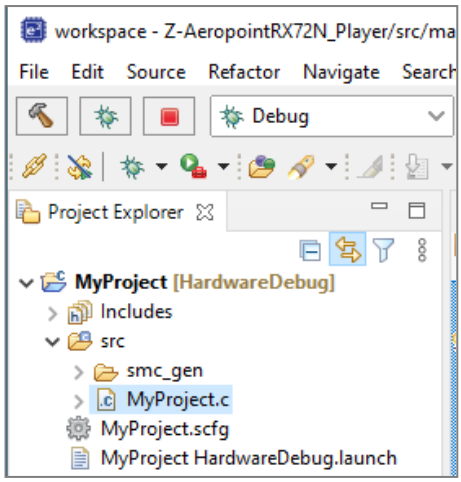


Fig. 60 "MyProject"

4.3.2 Setting up "QE for Display"

Next, set up "QE for Display".

Select "Main Menu" -> "Renesas Views" -> "Renesas QE" -> "LCD Main RX (QE)".

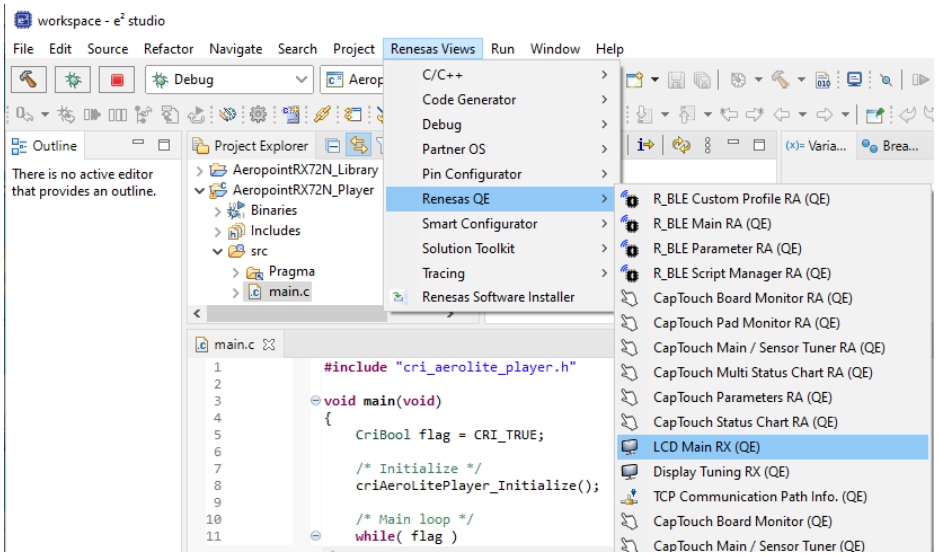


Fig. 61 Start "QE for Display"

After the "LCD Main RX (QE)" setting window is opened, follow the steps in the workflow diagram to configure the settings. First, proceed to "1. Preparation" procedure. In the "Select a project" pull-down, select the project you have just created.

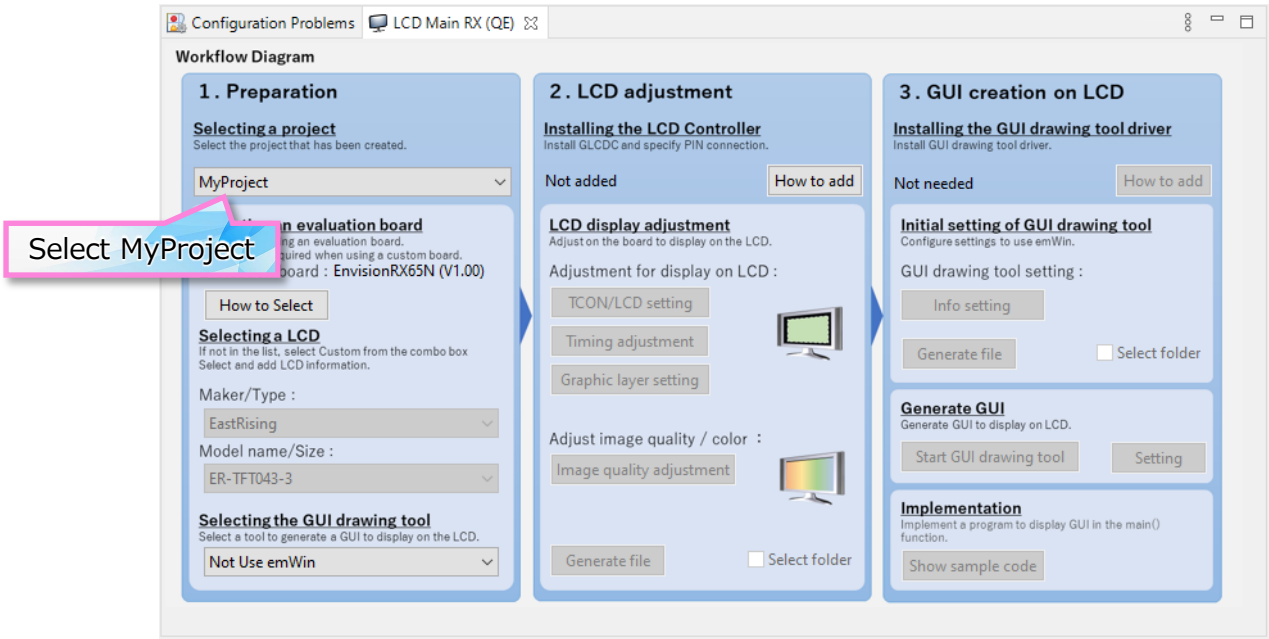


Fig. 62 Select project

The settings of the custom board and MCU (in this case, EnvisionRX65N) configured in the project will be automatically read and LCD-related information, etc. will be set. There are some settings that need to be done manually, so be sure to click the "How to select" button and check the help.

This time, you need to change the clock in the "Smart Configurator".

Double-click the "MyProject.scfg" file in the "Project Explorer" tab to open the "Smart Configurator". In that window, select the "Clocks" tab and set the "Frequency Multiplication" to "x20.0".

Finally, click on the code generation icon.

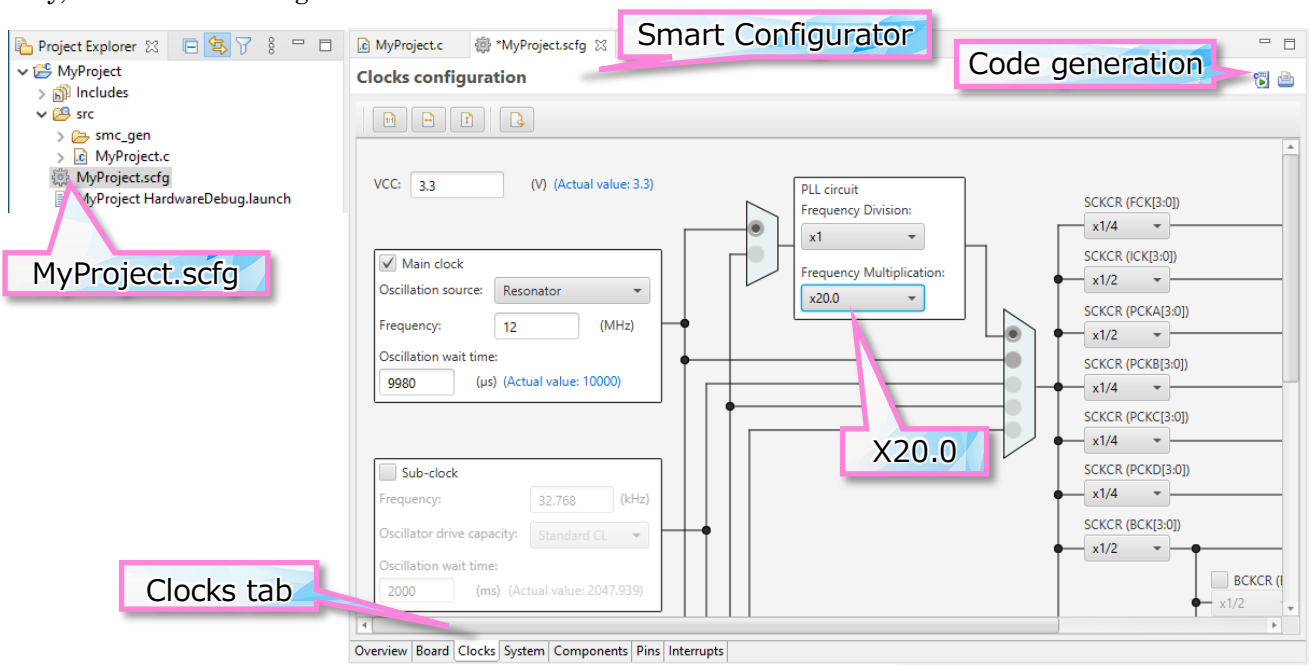


Fig. 63 Change "Frequency Multiplication"

Select "Use Aeropoint GUI" under "Selecting the GUI Drawing Tool". You are done "1. Preparation".

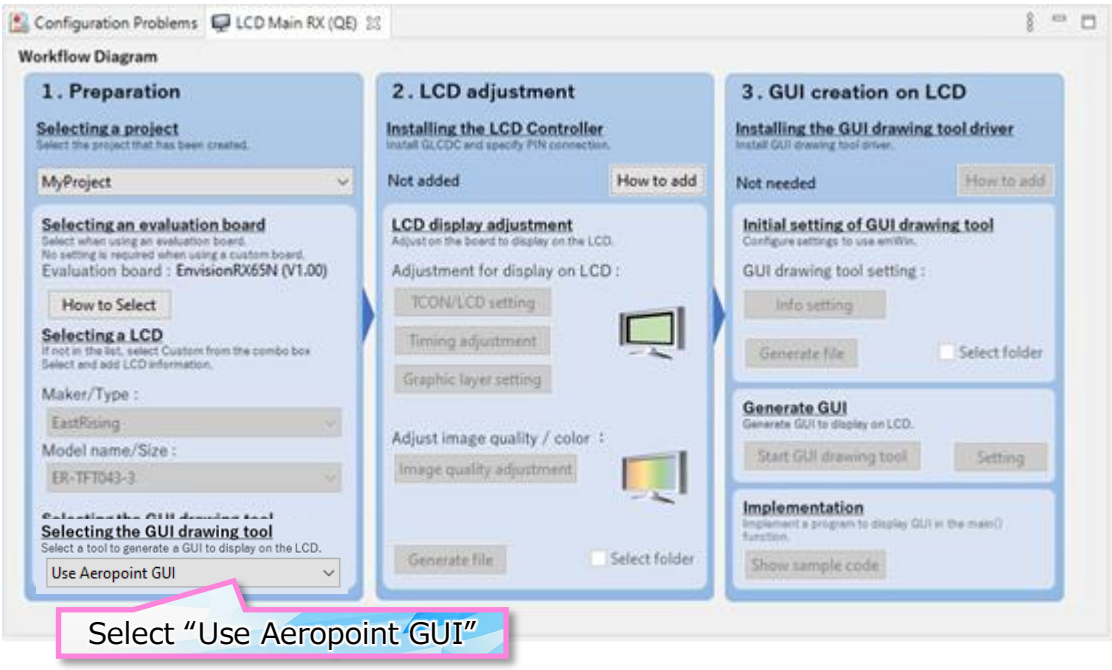


Fig. 64 select "Use Aeropoint GUI"

Next, proceed to "2. LCD adjustment". If you read the help that appears under "How to add", you will see that you need to add the GLCDC component.

Select the "Components" tab from the "Smart Configurator" window and click on the add component icon.

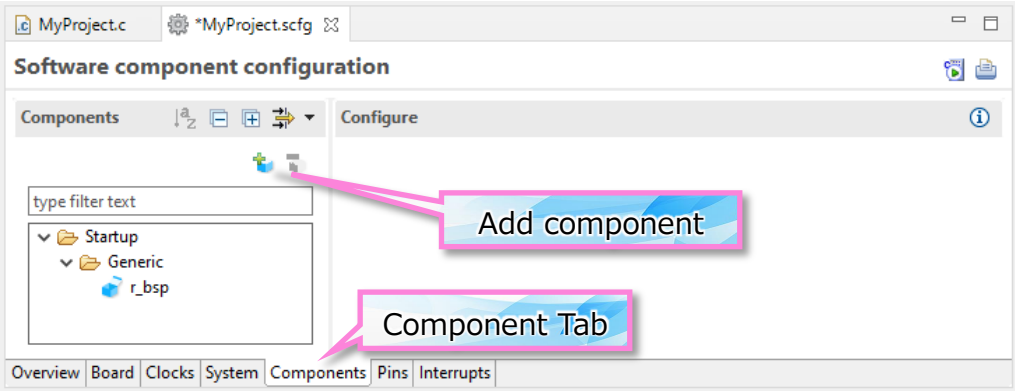


Fig. 65 Add component

When the "Software Component Selection" dialog box opens, select "Graphics LCD Controller Module" (GLCDC) and click "Finish"

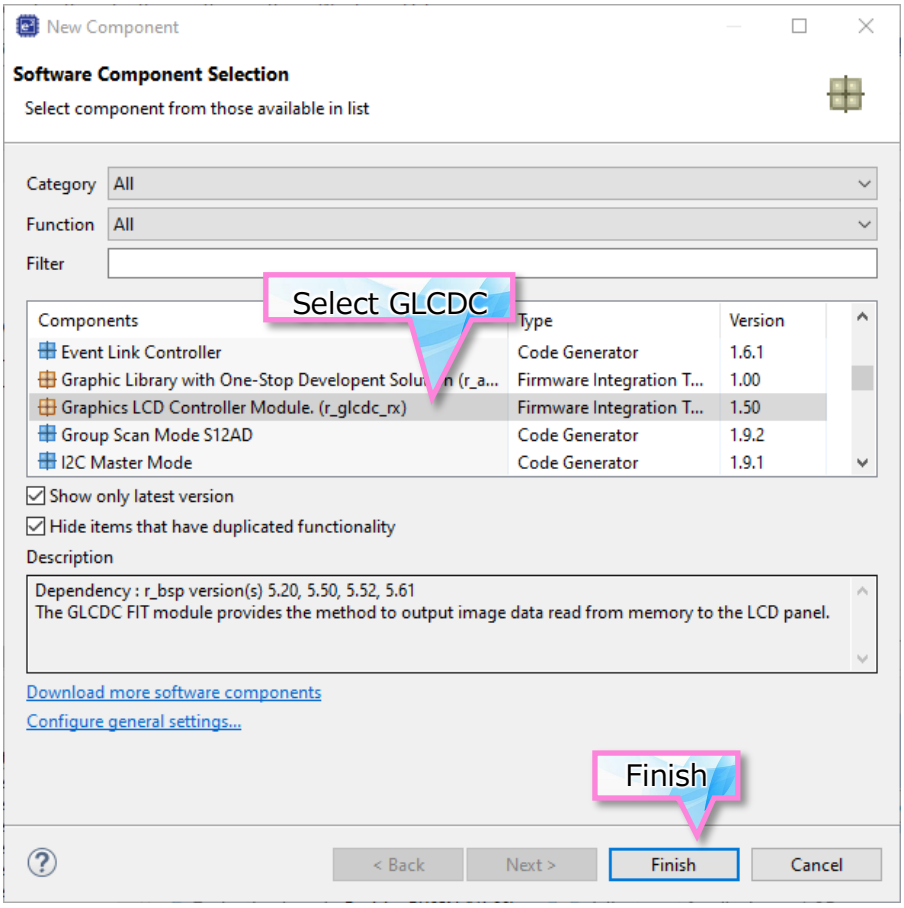


Fig. 66 Add GLCDC

"Graphics LCD Controller Module" (GLCDC) will be added to the "Smart Configurator" window, and click the code generation icon.

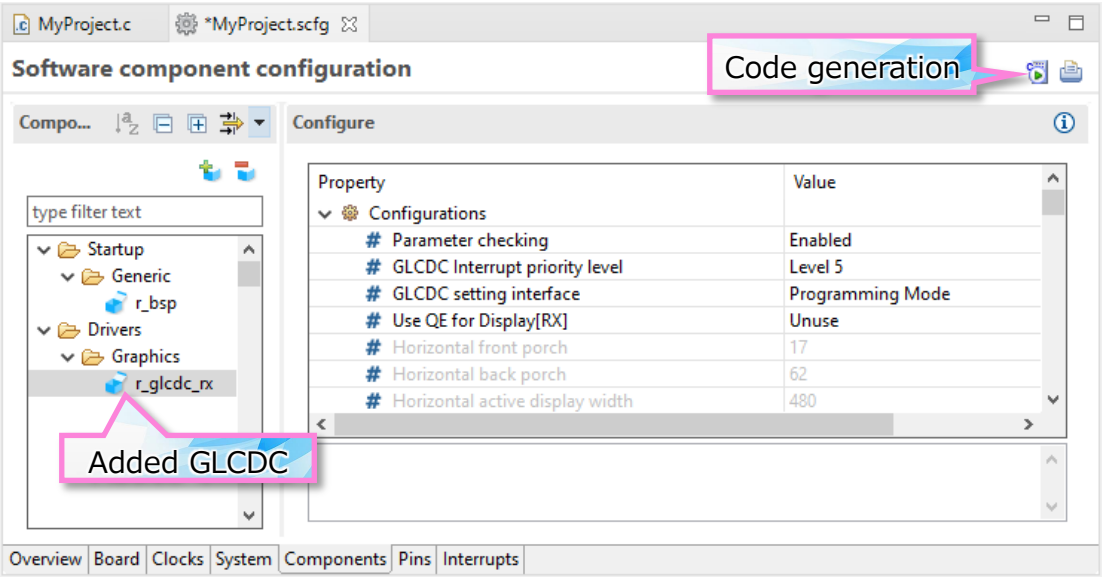


Fig. 67 Generate code

The LCD controller has been "Added" to the QE for Display window.
The display adjustment of the LCD is also set automatically, but be sure to open and check it once to make sure the settings are valid. If you are using other LCDs on your board, enter the settings that match the specifications of each LCD.

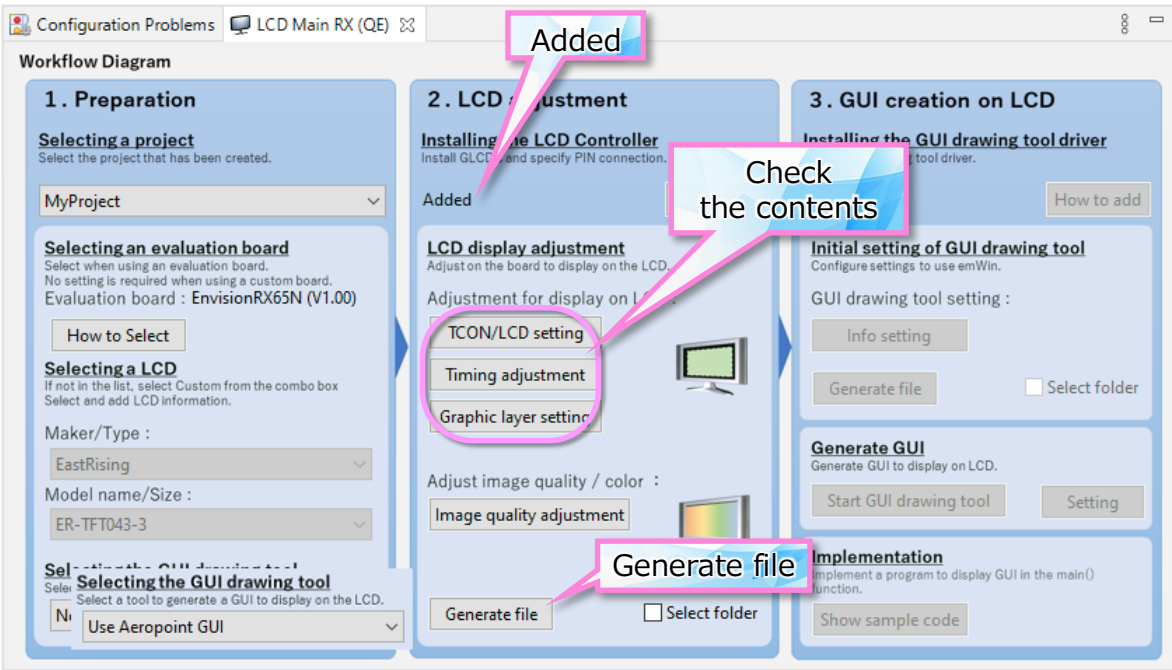


Fig. 68 Output LCD settings

Finally, proceed to "3. GUI creation on LCD". In the "Smart Configurator" window, select "Components" tab and click on the "Add component" button.

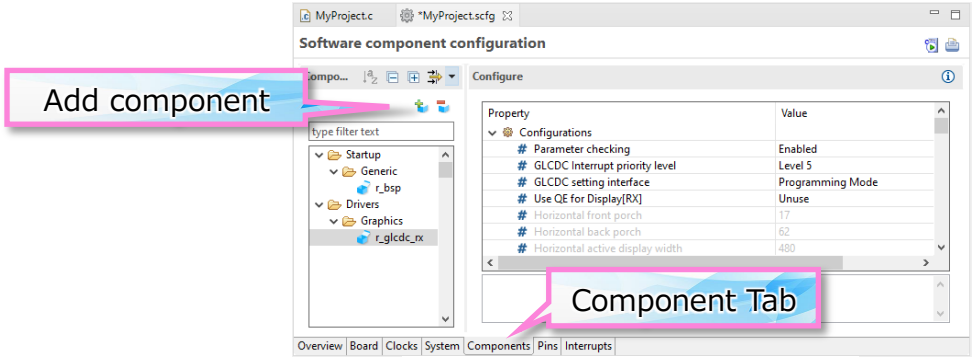


Fig. 69 Add component

Since there are so many components, enter the keyword "aero" in the "Filter" field to narrow down the list, then select Aeropoint (r_aeropoint_n) and click "Finish" button.

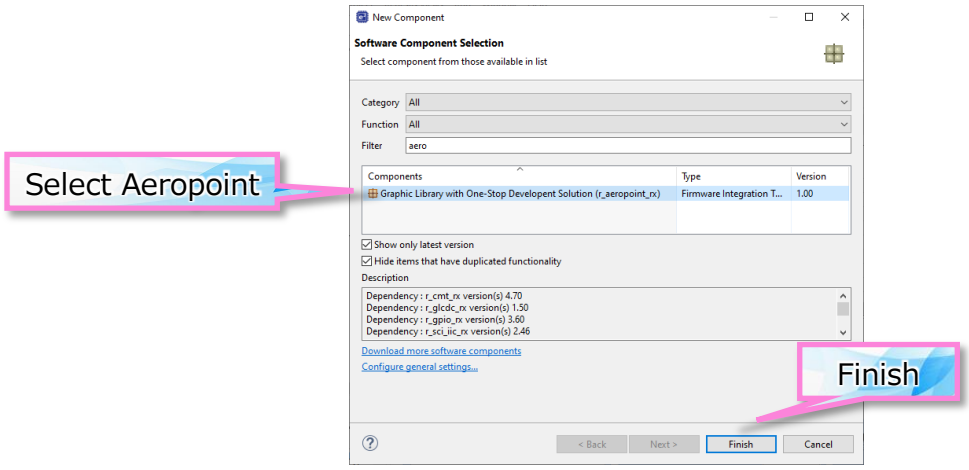


Fig. 70 Add Aeropoint

Aeropoint and its associated components will be added to the tree. Since Aeropoint uses the touch mechanism, select its component IIC (r_scr_iic_rx) and set CH6 to "Supported" by clicking in the "Configure" window. After the configuration is complete, click the code generation icon.

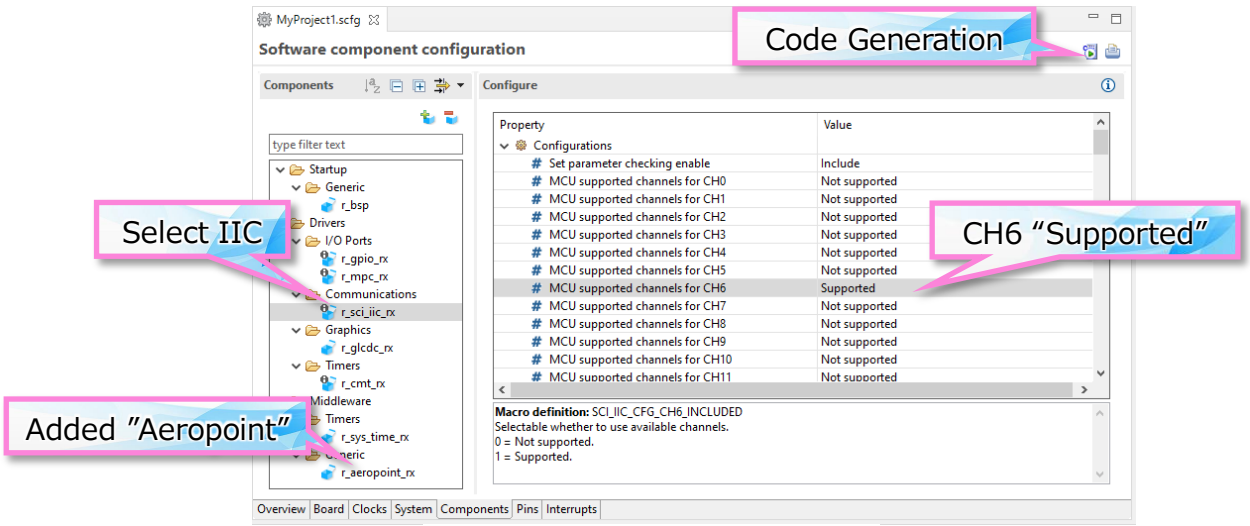


Fig. 71 Setting up the IIC

In the QE for Display window, "Added" is displayed in the "GUI creation on LCD" section.

Click "info setting" to configure Aeropoint.

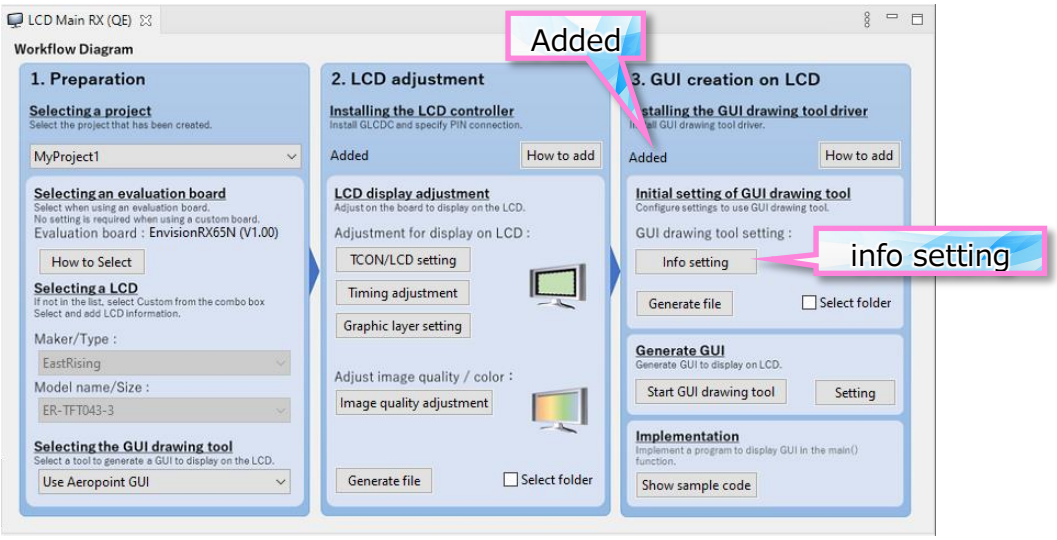


Fig. 72 info setting of "Aeropoint"

Dialog box will appear.

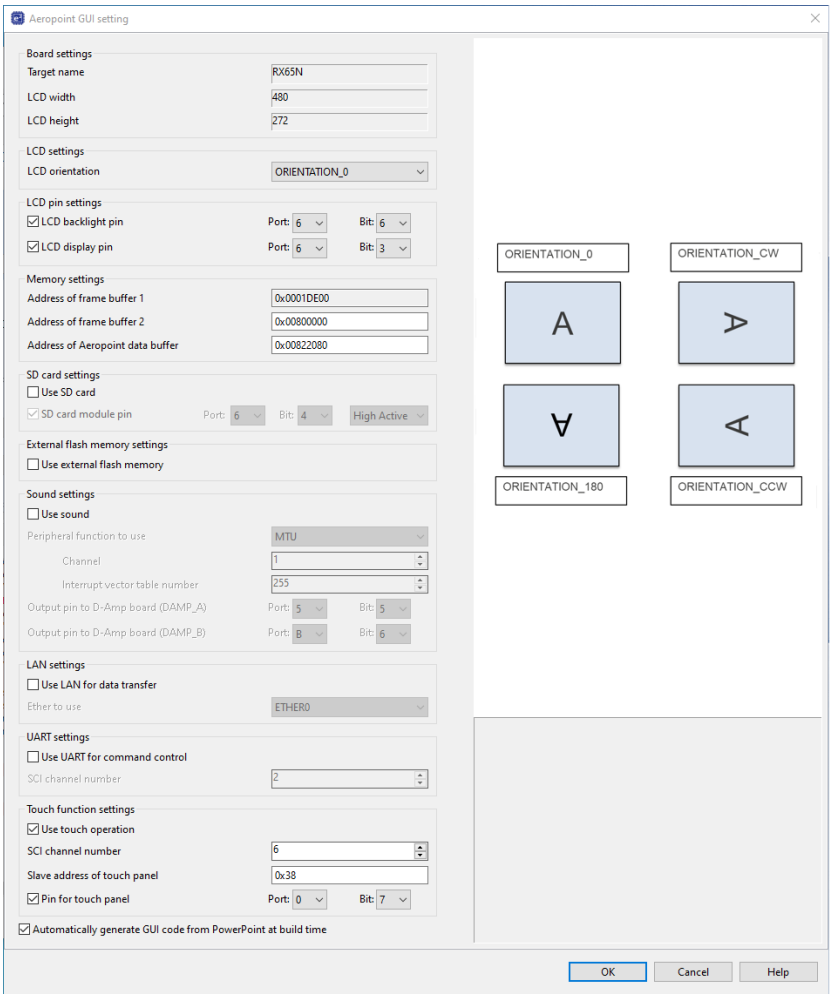


Fig. 73 dialog of "Aeropoint GUI setting"

- The "board setting" is already set.
- In "LCD settings", specify the orientation of the LCD.
- In "Memory settings", specify the address of the frame buffer. The frame buffer must have an area that matches the size of the screen.

  Ex. 480×272 = 130,560 = 0x1FE00

The following settings specify the functions that Aeropoint will use.

- SD card settings
- External memory settings
  Store screens, sounds, movies, fonts, etc. used by the Aeropoint runtime on SD cards or serial flash. Once you have checked the one you want to use, uncheck the other one.
- Sound settings
  Check box when using the D-Amp Driver for sound playback.
  A dedicated circuit is required for sound playback using the D-Amp Driver.
- LAN settings
  Check this box if you want to communicate with the Aeropoint toolkit via LAN. You can write data to SD card, serial flash, etc. and communicate commands for debugging.
- UART settings
  Check this box if you want to use Aeropoint runtime for command communication via UART.
- Touch function settings
  If you do not want to use the touch function, uncheck the box. The "SCI channel number" is set to "6" because IIC CH6 is set to "Supported" in "Software Components".

Each function (module) of Aeropoint depends on the FIT module, as shown in the figure below. If you want to add Aeropoint functionality in "Aeropoint GUI setting", please add the associated components and generate the code. This time, leave the default settings and click "OK" to close the dialog.

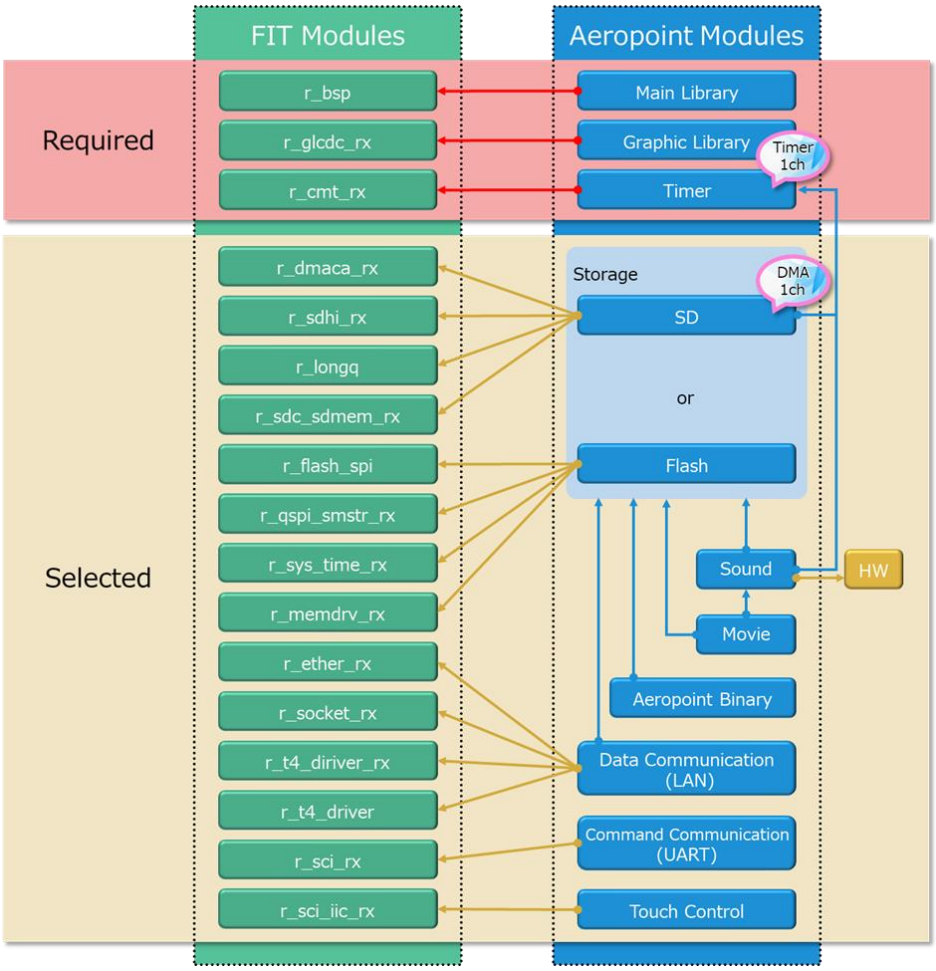


Fig. 74 Dependency between FIT module and Aeropoint module

Click "File Output" in the QE for Display window to output the code of the component tied to Aeropoint.

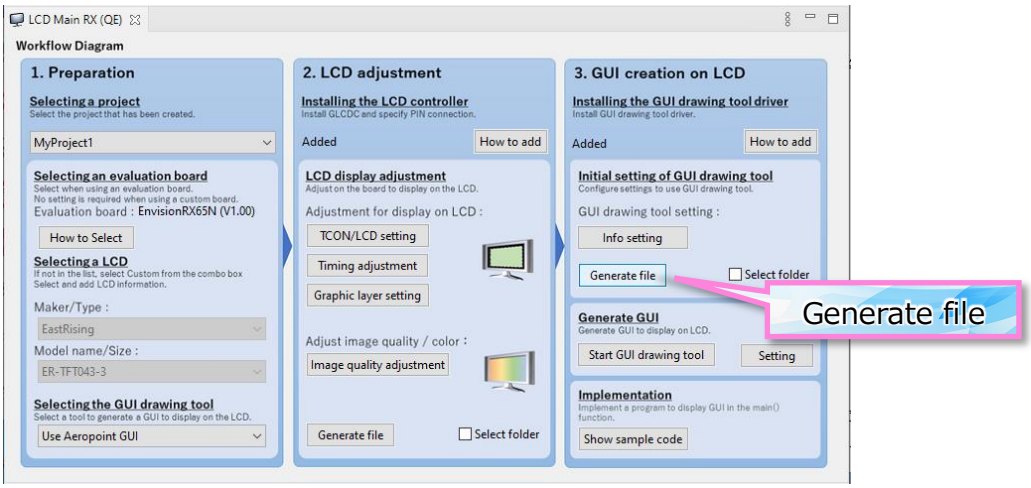


Fig. 75 "Generate file"

This completes the setup of the project.

4.4 Create GUI

Next, proceed to "Create GUI".

If you do not have the Aeropoint toolkit on your PC, click "Setting" and follow the wizard to install it.

After the installation, click "Start GUI drawing tool" to start the tool.

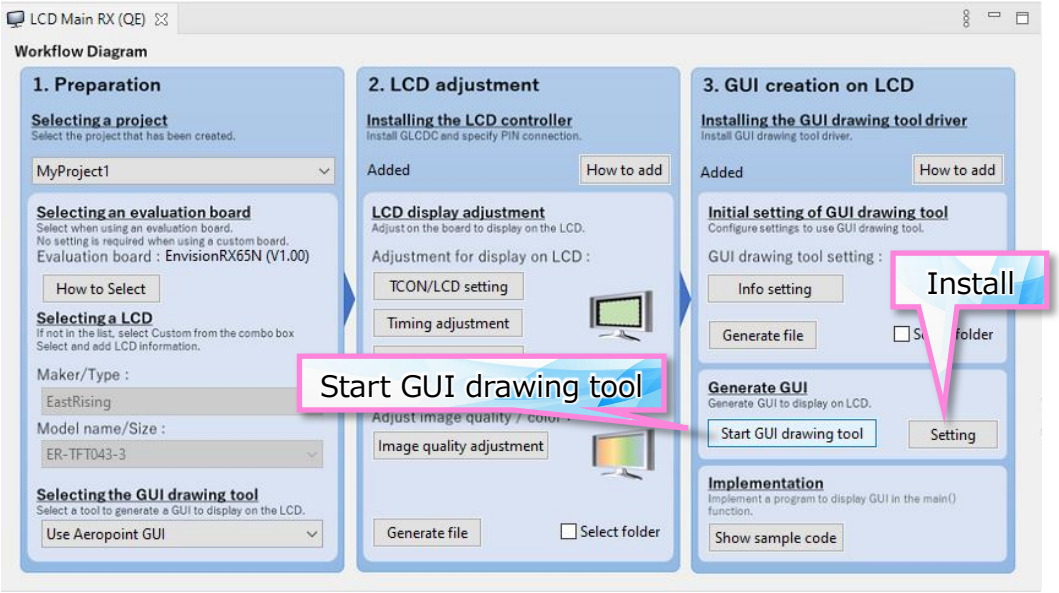


Fig. 76 "Generate file"

PowerPoint® will be launched with the Aeropoint add-in added.

Draw a GUI of your choice, which can be anything you want.

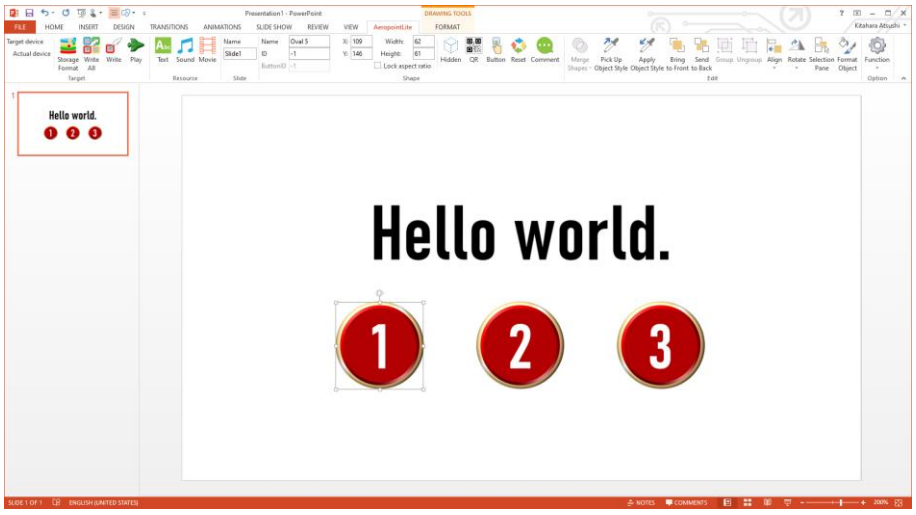


Fig. 77 Draw a GUI

When you are done creating the GUI, overwrite and save the file. The PPTX file you created here will be automatically added to "MyProject" and managed by e²studio.

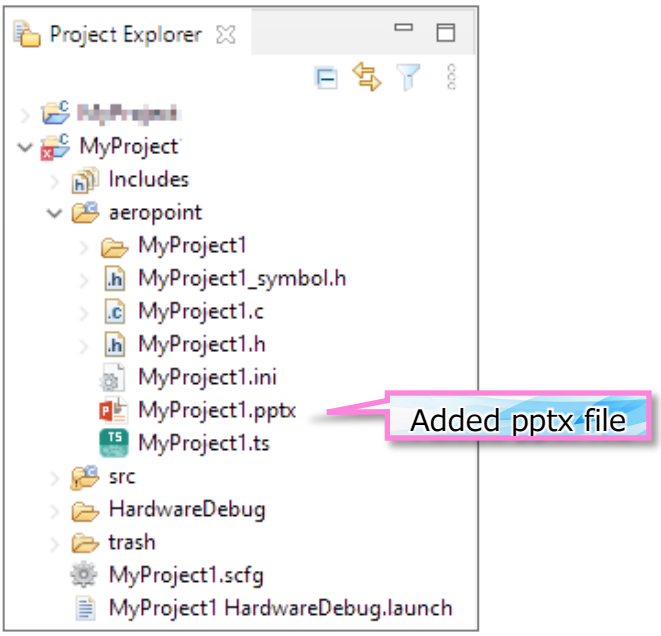


Fig. 78 added pptx file

4.5 Library

To use the FIT kit runtime, you need to link the Aeropoint runtime library to your project in addition to the FIT module. Please use the library in the "lib" folder of the generated FIT kit according to the combination of devices and compilers.

Group microcontrollers	compiler	Aeropoint library
RX65	CCRX V2	libcri_Atom_Multiplayer_RXv2_CCRX.lib libcri_AeropointLite_RXv2_CCRX.lib
RX72	CCRX V2	libcri_Atom_Multiplayer_RXv2_CCRX.lib libcri_AeropointLite_RXv2_CCRX.lib
	CCRX V3	libcri_Atom_Multiplayer_RXv3_CCRX.lib libcri_AeropointLite_RXv3_CCRX.lib

Fig. 79 library

4.6 Create and Execute Program

"QE for Display" has a function to display sample code that matches the project you have created. If you want to see the screen first, click "Show sample code" to display the sample code.

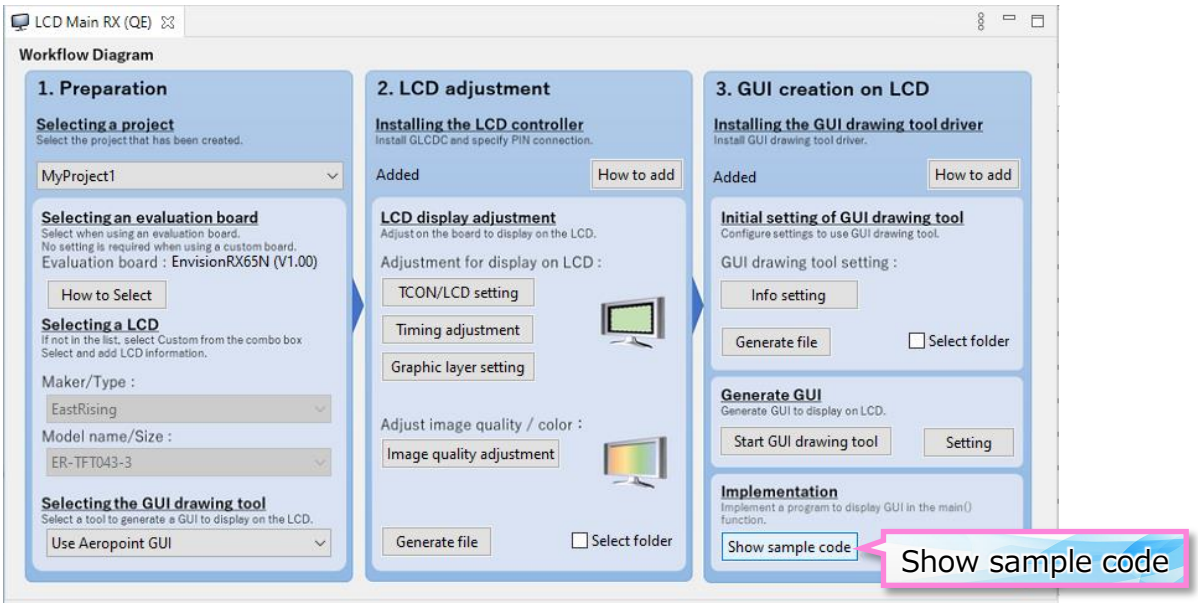


Fig. 80 "Show sample code"

When the sample code is displayed, click "Copy to the Clipboard".

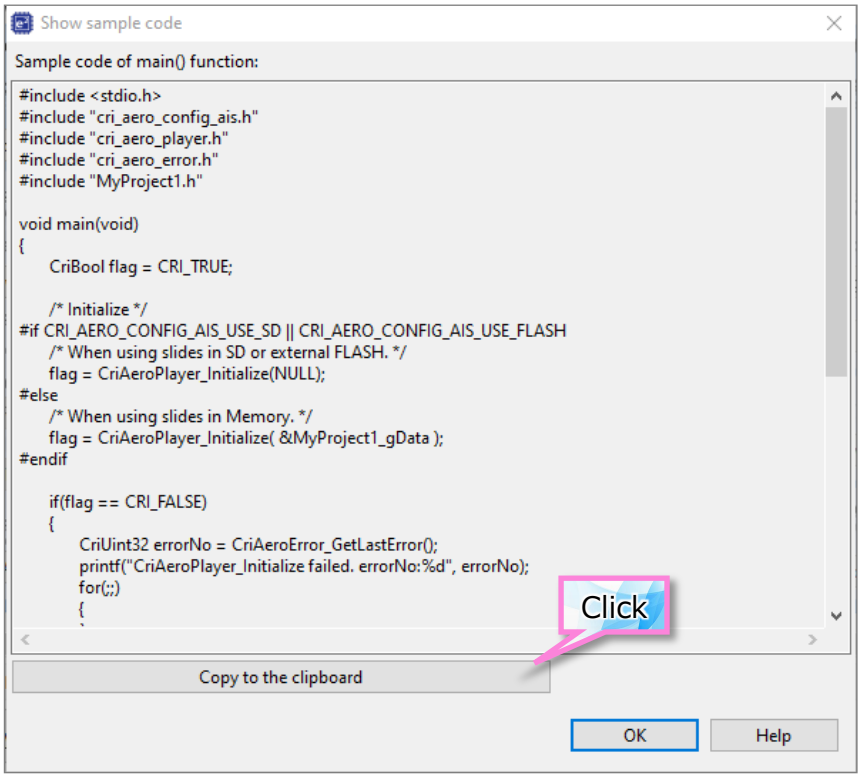


Fig. 81 Copy the sample code

Paste it into "MyProject.c", the main program.

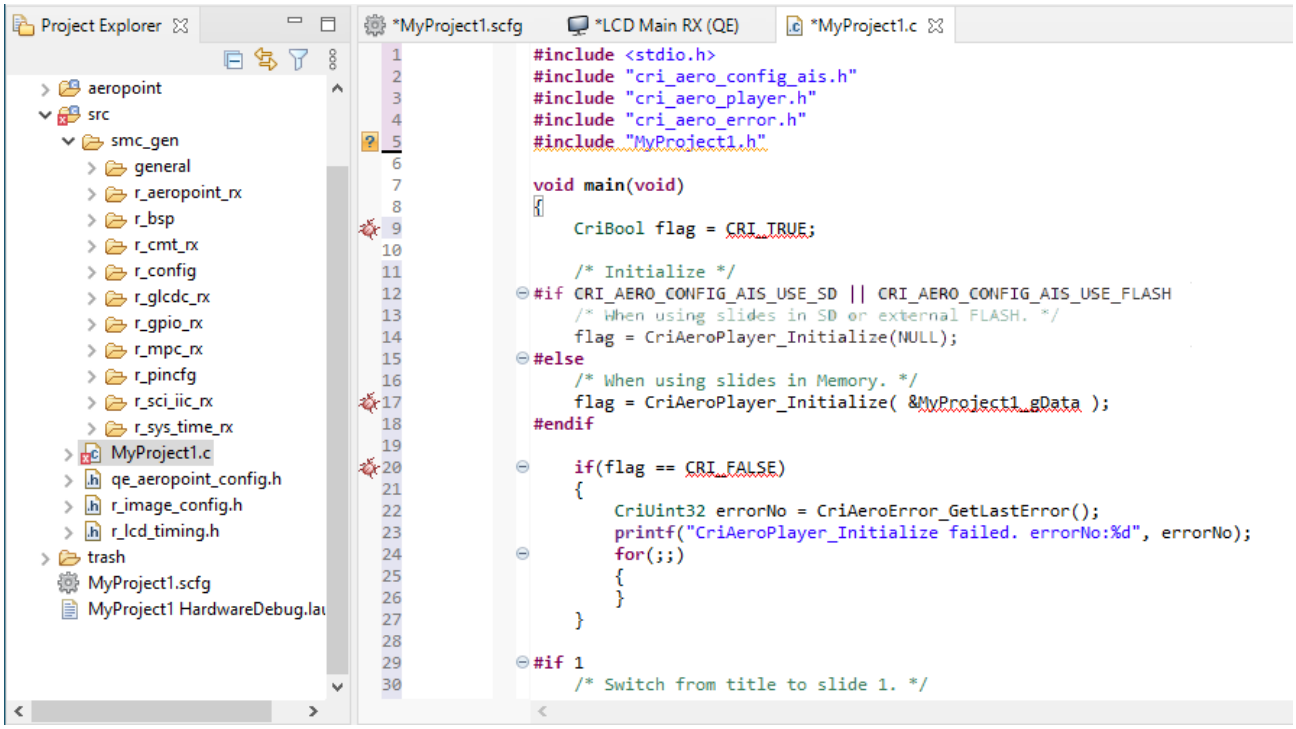


Fig. 82 Pasted screen display program

When you build and run the program, you will see the screen you just created.

5 Aeropoint toolkit features

Describes the features of the Aeropoint toolkit, PowerPoint® add-in.

5.1 "Target" group

The "Target" group can connect to the actual device via "Connector" tool and write data to the SD card, etc.

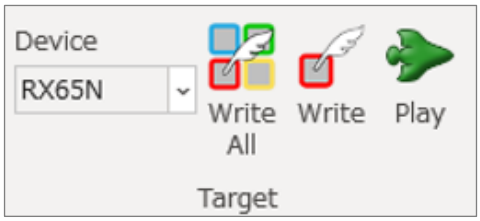


Fig. 83 "Target" group

5.1.1 "Write All"

"Write All " will export the data of all pages and write the data to the connected SD card of the actual device. This "Export" function works automatically, such as the undo buffer is cleared, thus we recommend you to save the data frequently.

However, if set to "Hidden Slide", the selected slide will not be exported and the next slide will be shown. Move these

"Hidden Slide" to the end to prevent page number mismatch.

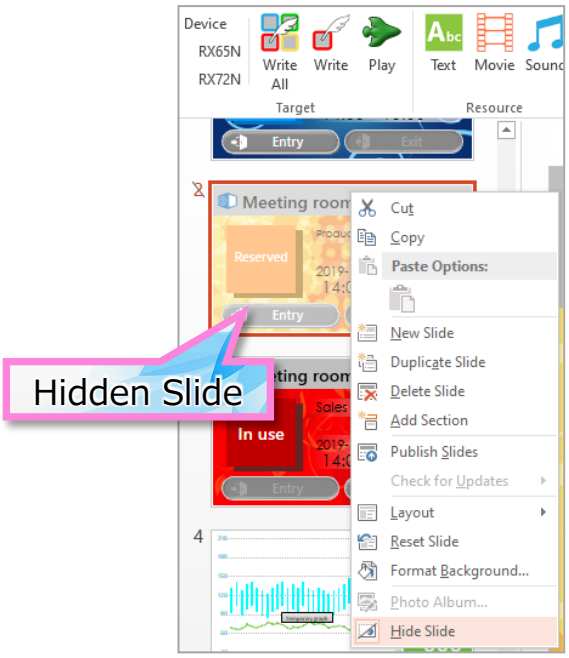


Fig. 84 Hidden Slide

5.1.2 "Write"

"Write " writes only the data of the selected slide (ONE) to the SD card. Multiple selection is disabled.

5.1.3 "Play"

"Play " activates the "Connector" tool and sends a command to the actual device to display the first slide.

"Play" does not write the slide data to the SD card. If you want to update the slide data of the actual device, use "Write All" or "Write".

5.2 "Resource" group

"Resource" group handles external data. Target data will be captured in the PowerPoint® file.

Fig. 85 "Resource" group

5.2.1 "Text"

"Text" imports the font data (AC1FC format file) created by the tool mentioned in Chapter "9 Make Font" to PowerPoint®, and will write to the SD card if necessary.

Press "Text" to show the following dialog. Drop the font data in the dialog and click "OK" to import the font into PowerPoint®. Select the checkbox on the left side and click "Transfer selected Items" to write the selected font data to the SD card in the actual device using LAN connection.

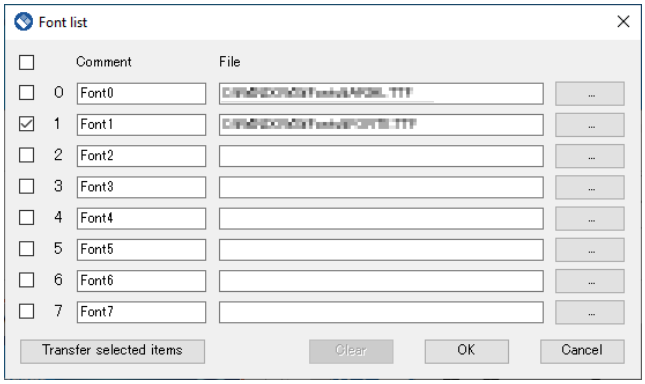


Fig. 86 "Font list" dialog

Since the size of the font, video, and sound data are big, it takes time when writing. The "Connector" may stop responding for a certain period of time, but be patient till the transfer is completed. If the transfer is done, the green LED on the LAN port of the actual device will flash. While if the LED remains ON or OFF, a transfer error has occurred. Try the transfer again.

5.2.2 "Movie"

"Movie " imports MP4 format files into PowerPoint@, and you can write to the actual SD card.

Click "Movie" to show the following dialog. Drop the video data here and click "OK" to import the movie into PowerPoint@.
Select the checkbox on the left and click "Transfer selected Items" to write the selected video data to the SD card in the actual device using LAN connection. Import to PowerPoint@ as MP4 format and write to the actual device.
MP4 format file will be imported directly to the memory of the actual device, thus do not use large size videos, but smaller ones.

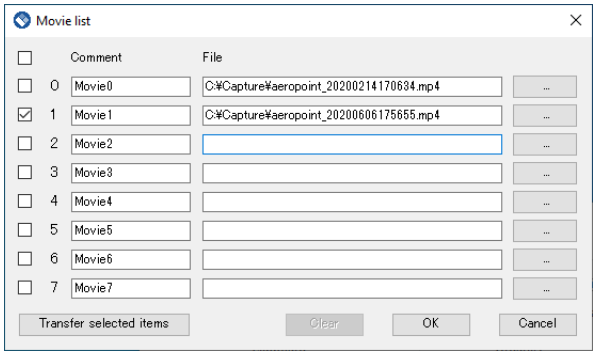


Fig. 87 "Movie list" dialog

Recommended video format:

- File format
  - MP4 only
- Length
  - Up to 30 seconds
- Resolution
  - 480 x 272 pixel
    - ※ For RX72N Envision Kit
- Frame rate
  - Over 15 fps
    - ※ Video is fixed at 15fps on the actual device.
- Sound
  - Monaural, sampling frequency 24kHz

5.2.3 "Sound"

"Sound " imports a WAV format or MP3 format file into PowerPoint® and write to the SD card.

Click "Sound" to show the following dialog. Drop the sound data here and click "OK" to import the sound into PowerPoint®. Click "Transfer" to write the sound data to the SD card in the actual device.

Sound data is written to the SD card in batch.

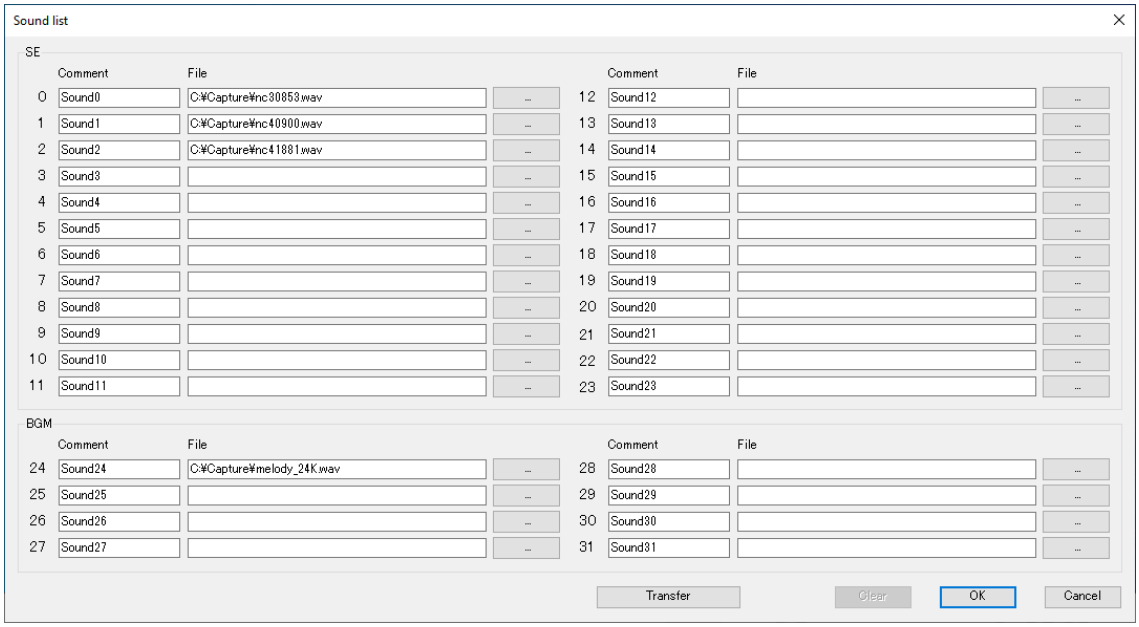


Fig. 88 "Sound list" dialog

Recommended sound format

- File format
  - WAV , MP3
- Length
  - SE : Up to 10 seconds
  - BGM : Up to 3 minutes
- Sampling rate
  - Monaural, sampling frequency 24kHz

5.3 "Slide" group

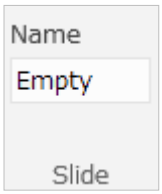


Fig. 89 "Slide" group

5.3.1 Slide Name

You can rename the slide. Characters that can be used are "half-width alphanumeric characters + '_' (underscore)" only.

5.4 "Shape" group

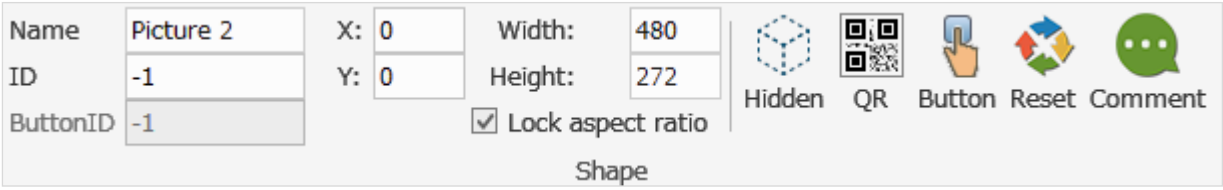


Fig. 90 "Shape" group

5.4.1 Shape Name

Select one shape and rename it. Characters that can be used are "half-width alphanumeric characters +'_' (underscore)" only.

5.4.2 ID

Shape ID is "-1 (disabled)" by default. The invalid shape is grouped with other similar shapes and is displayed as the last background on the actual device. Therefore, the display may differ due to the context of PowerPoint®.

Enter a value of 0 or more to create an independent shape. It can be controlled by commands and programs.

5.4.3 Button ID

If a button is set to the selected shape, button ID will be displayed. You can also change the ID.

5.4.4 X / Y / Width / Height / Lock aspect ratio

Here you will find the information about the placement and size of the selected shape. You can change the position and size of the shape by changing the value. Select the "lock aspect ratio" checkbox to change the width and height while maintaining the aspect ratio.

5.4.5 Hidden

"Hidden " will not display the selected shape on the actual device. Since the data is not exported, the information of this shape will not be included in the output data. Click "Hidden" again to return to visible status.

You can hide layouts that do not need to be displayed on the actual device, and shapes for explanation, etc.

5.4.6 QR

Select a square shape and click "QR " to open the QR code setting dialog. After entering the data you want to display as a QR code, click "OK".

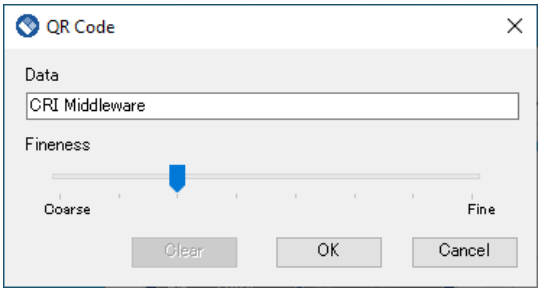


Fig. 91 "QR Code" setting dialog

"QR Code" remains a square shape on PowerPoint®, but it will be shown on an actual device.

5.4.7 Button

"Button " sets the behavior of the selected shape as a button.

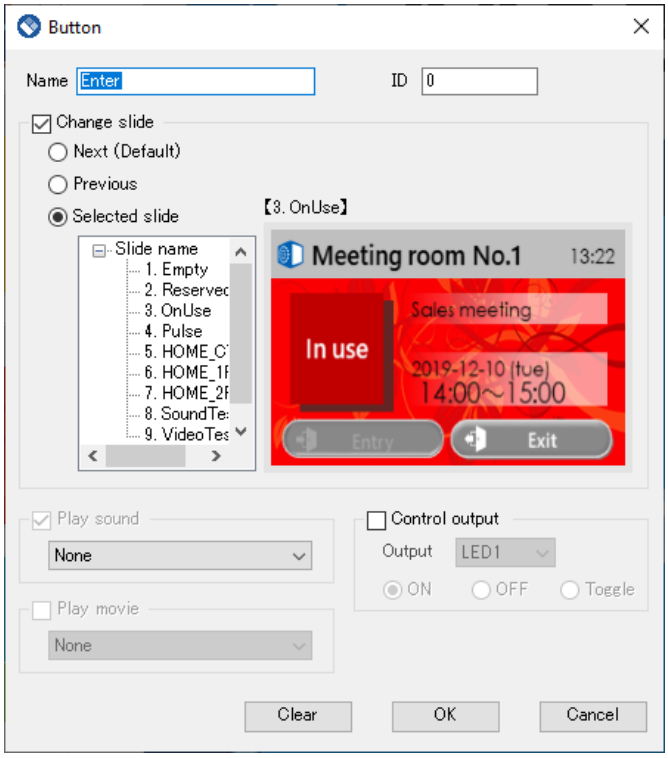


Fig. 92 Button Dialog

You can rename the "Button" dialog by using only "alphanumeric characters +'_' (underscore)".
Enter a value of 0 or more in "ID" to receive the callback from the program when touched.

Set "Change slide" to switch from the previous/next slide or specified slide when the button is touched.
Set "Play sound" or "Play movie" to play the specified sound effect and movie when the button is touched.
Set "Control output" is set to change the lighting status of the LED on the D-Amp board using the I/O test.

5.4.8 Reset

"Reset " cancels settings such as QR code and buttons.

5.4.9 Comment

"Comment " allows you to set a comment for the selected shape. When you select a shape with a comment, its content is displayed in the "Comment" dialog box.

5.5 "Edit" group

Use the "Edit" function provided by PowerPoint® that are useful for creating GUIs.

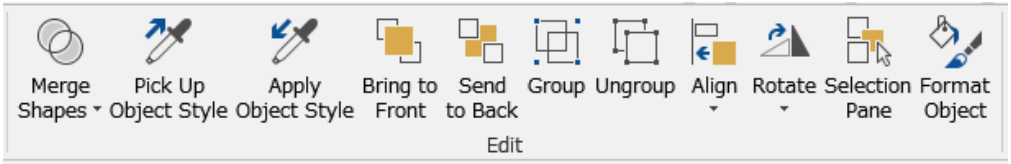


Fig. 93 Edit group

5.5.1 Merge Shapes

If you select more than one shape (without text), you can use the "Merge Shape" menu to create various joining patterns, as shown below.

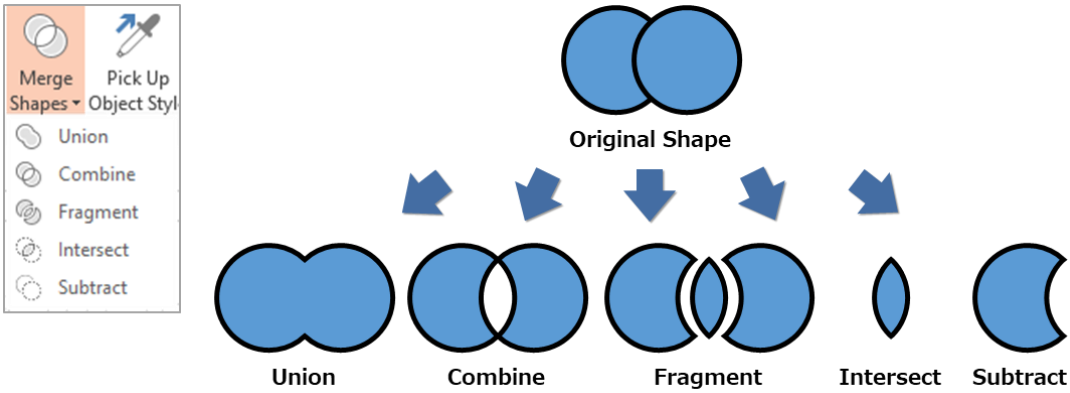


Fig. 94 Various merge patterns you can create by this function in PowerPoint®

"Subtract" is often used. When selecting multiple shapes, you can subtract the shapes you selected after the first selected shape.

Fig. 95 Example using "Subtract"

By using "Merge Shapes", you can create unique UI shapes other than the standardized PowerPoint® shapes.

Fig. 96 Original arrow key with Enter key

5.5.2 Free form & Subtract

This section describes how to make the arrow key described in the previous section. The figure made from the circle using "Subtract" is one figure with four keys. Therefore, you cannot operate the individual keys as they are.

You create a free form shape for subtraction, overlay it on the shape above, and use "Subtract" to cut out only where you need it.

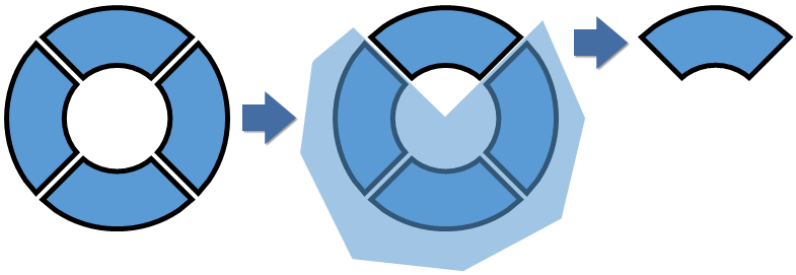


Fig. 97 Free form & Subtract

You can create four keys by repeating this procedure four times.

5.5.3 Copy & Apply Object Style

In PowerPoint®, various settings can be made to the shapes, such as changing color and text format, adding lines, light effects, etc. However, it takes time to apply the created settings to multiple shapes. To make everything easy, you can use the "Pick Up Object Style" and "Apply Object Style" to copy and apply the settings.

Select the source shape and click "Pick Up Object Style" (or [ctrl] + [shift] + [C]), select another shape and click "Apply Object Style" (or [ctrl] + [shift] + [V]) to apply. It has no effects on the shape or the text inside the shape, only the format setting will be applied.

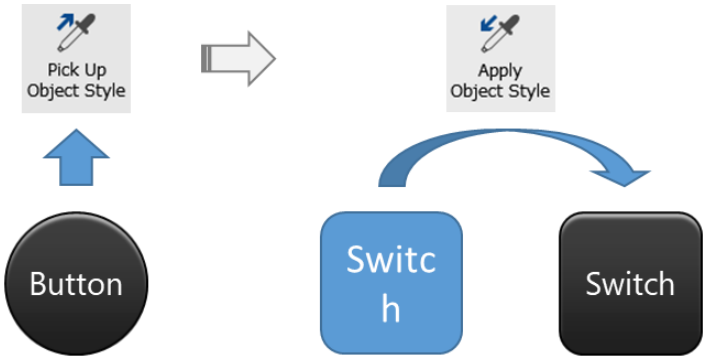


Fig. 98 Copy & Apply "Object Style"

5.5.4 Bring to Front / Send to Back

When combining several shapes, the shape to be created/pasted is always placed in the foreground, so it is necessary to adjust the context. You use "Bring to Front" to bring the selected shape to the front and "Send to Back" to send it to the back. By selecting multiple shapes, you can also adjust all in one click.

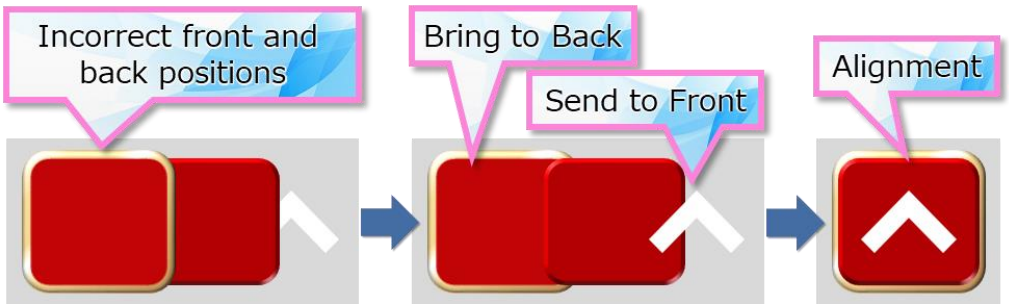


Fig. 99 Adjusting context

5.5.5 Group / Ungroup

You can use "Group" to combine multiple shapes into one. You can also ungroup at any time using "Ungroup".

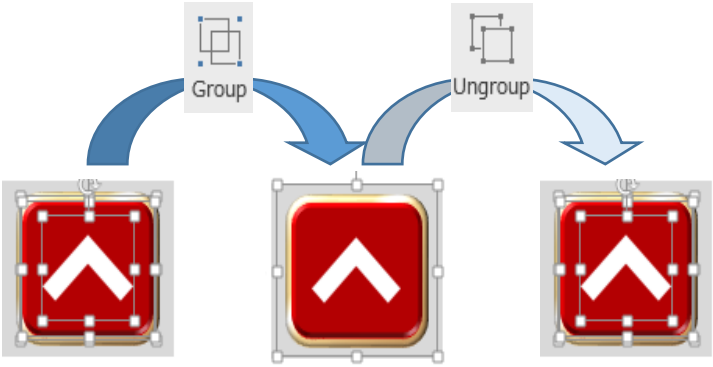


Fig. 100 Group / Ungroup

5.5.6 Align

When copy and paste the shapes, you may need to reposition them. Rather than placing them manually, use this "Align" function to easily adjust the position of the shapes. You can align the shapes vertically or horizontally as below.

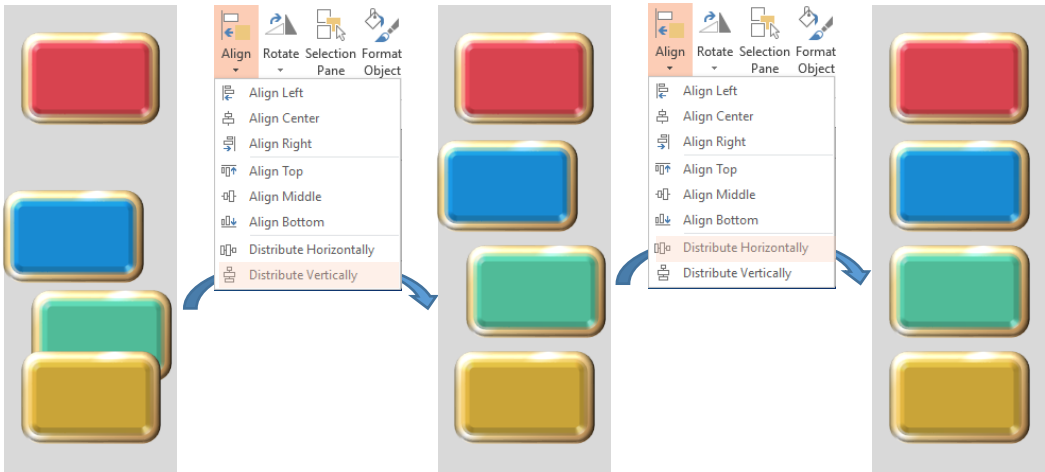


Fig. 101 Align

5.5.7 Rotate

Use "Rotate" to rotate or flip.

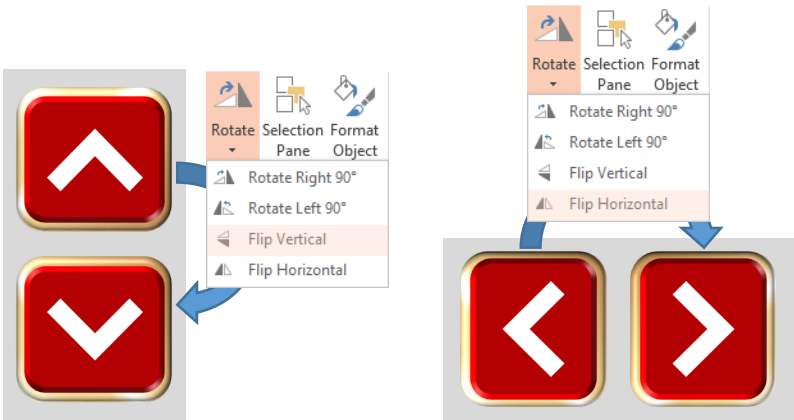


Fig. 102 Rotate

5.5.8 Selection Pane

Use the "Selection" function to manage the context and names of shapes collectively.

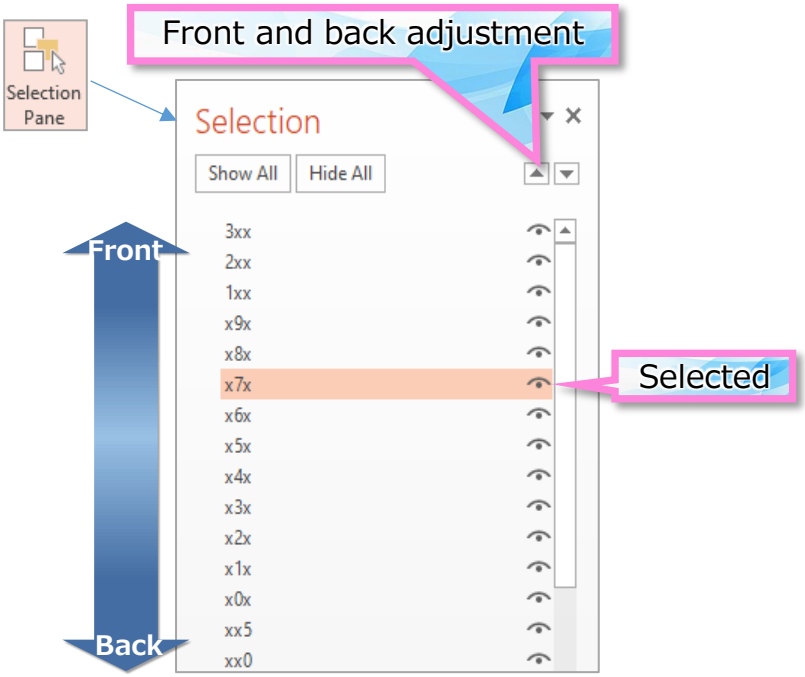


Fig. 103 Selection

You can change the name by clicking on the name of the shape. Names will reset if the shapes are grouped or ungrouped. Therefore, we recommend to adjust the context and name at the last minute.

When programming using Aeropoint runtime, the shape using the name set here may be referenced. Only "alphanumeric characters" can be used. Note that using Japanese characters will cause an error.

5.5.9 Format Object

You can add various effects to the shape with the following "Format Object" function.

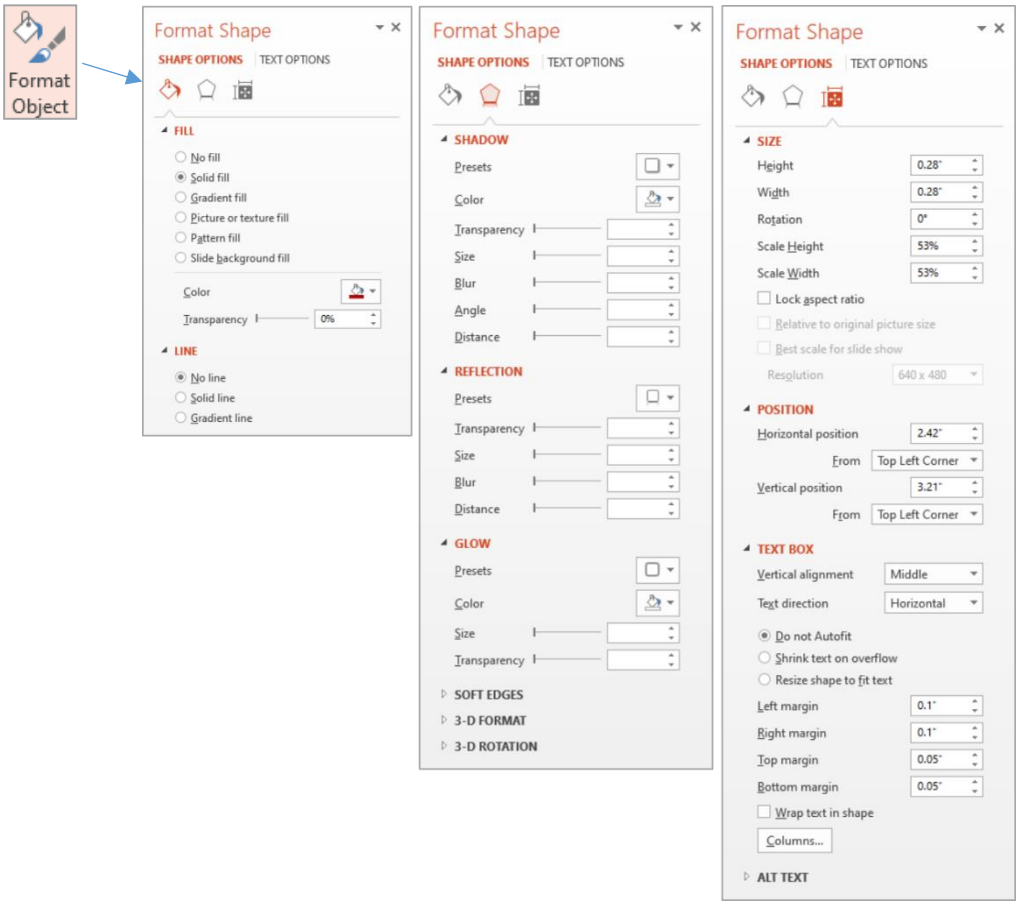


Fig. 104 Format Object

We will not mention every function, but below is an example of adding gradient stops to the border to give it a "luxury feeling"

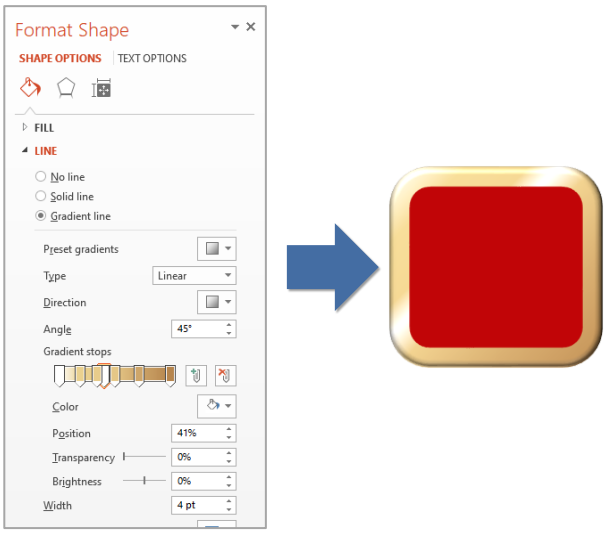


Fig. 105 Example of how to add color to the shape

5.6 "Option" group

The "Option" group is a collection of auxiliary features of the Aeropoint toolkit.

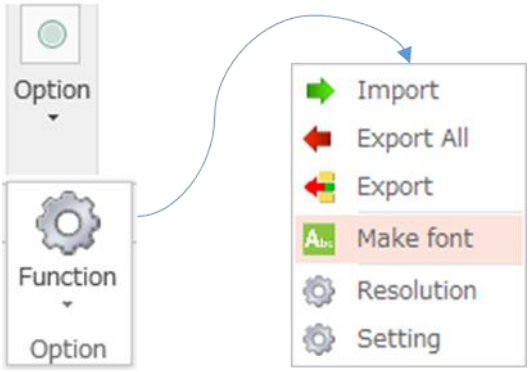


Fig. 106 "Option" group

5.6.1 Import

You can use "Import" to import the images exported from Photoshop as slides. The function to export from Photoshop is an optional feature. To use it, please contact CRI (refer to Chapter "14 Contact us").

5.6.2 Export All / Export

"Export All" saves all slide information as data. "Export" saves the current slide information as data. You can write the saved data (in AEB format) to the SD card using the "Connector" tool.

"Export All" also saves a header file that summarizes the symbol information of the shapes set on the slide. The output symbol information is the shape ID and button ID set for each shape.
The symbol information in your programming will be saved when creating your Aeropoint application.

```
/* Sample_Remocon Slide Main */
#define Sample_Remocon_Main_BackGround_ShapeID                    (-1)
#define Sample_Remocon_Main_Group_7_ShapeID                       (-1)
#define Sample_Remocon_Main_Group_7_ButtonID                      (6)
#define Sample_Remocon_Main_Group_139_ShapeID                     (-1)
#define Sample_Remocon_Main_Group_139_ButtonID                    (7)
#define Sample_Remocon_Main_Group_165_ShapeID                     (-1)
#define Sample_Remocon_Main_Group_165_ButtonID                    (0)
#define Sample_Remocon_Main_Group_170_ShapeID                     (-1)
#define Sample_Remocon_Main_Group_170_ButtonID                    (1)
#define Sample_Remocon_Main_Group_175_ShapeID                     (-1)
#define Sample_Remocon_Main_Group_175_ButtonID                    (2)
#define Sample_Remocon_Main_Group_180_ShapeID                     (-1)
#define Sample_Remocon_Main_Group_180_ButtonID                    (3)
#define Sample_Remocon_Main_Group_83_ShapeID                      (-1)
#define Sample_Remocon_Main_Group_83_ButtonID                     (4)
#define Sample_Remocon_Main_Group_89_ShapeID                      (-1)
#define Sample_Remocon_Main_Group 89_ButtonID                     (5)
#define Sample_Remocon_Main_xx0_ShapeID                           (0)
#define Sample_Remocon_Main_xx5_ShapeID                           (5)
#define Sample_Remocon_Main_x0x_ShapeID                           (10)
#define Sample_Remocon_Main_x1x_ShapeID                           (11)
#define Sample_Remocon_Main_x2x_ShapeID                           (12)
#define Sample_Remocon_Main_x3x_ShapeID                           (13)
#define Sample_Remocon_Main_x4x_ShapeID                           (14)
#define Sample_Remocon_Main_x5x_ShapeID                           (15)
#define Sample_Remocon_Main_x6x_ShapeID                           (16)
#define Sample_Remocon_Main_x7x_ShapeID                           (17)
#define Sample_Remocon_Main_x8x_ShapeID                           (18)
#define Sample_Remocon_Main_x9x_ShapeID                           (19)
#define Sample_Remocon_Main_1xx_ShapeID                           (100)
#define Sample_Remocon_Main_2xx_ShapeID                           (200)
#define Sample_Remocon_Main_3xx_ShapeID                           (300)
```

Fig. 107 Header file example

5.6.3 Create Font

"Create Font" starts the tool that creates font data.
Refer to Chapter "9 Make Font" for details.

5.6.4 Resolution

Use "Resolution" to set the resolution of PowerPoint® that is the same as the actual device.

- RX72N Envision Kit
  - Width : 480 pixels
  - Height : 272 pixels

5.6.5 Settings

"Settings" specifies the optimization of image data. No need to change the setting as almost optimal data will be output automatically.

6 Try it

In this chapter, we will introduce a prototype sample of how to create a new GUI for "Smart control for air conditioner".

Fig. 108 Prototype GUI design for reference

You may refer to this sample for inspiration to create your own unique GUI by Aeropoint.

6.1 Preparation

6.1.1 Create new

Create a new PowerPoint® file and save it as:

- Sample\GUI\Sample_Remocon.pptx

Remember to save and backup frequently during the GUI creation.

6.1.2 Change resolution

Use the Aeropoint tool to set PowerPoint® settings to match the actual resolution. Select the Aeropoint Add-in Menu> Options> Function> Resolution.

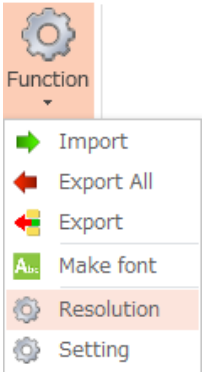


Fig. 109 Select "Resolution"

Change resolution to 480x272.

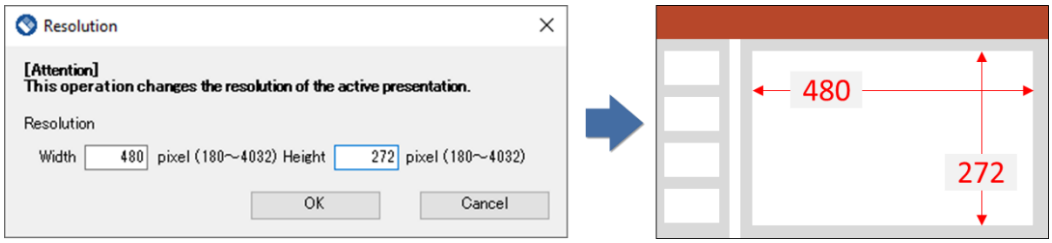


Fig. 110 Change resolution

6.2 Design your own GUI

Unlike the requirements and basic specifications that have be decided when starting product development, you may need to have several GUI designs for trial and error. Here are some suggestions from CRIWARE.

6.2.1 Create an overall picture

First of all, decide the overall picture of the GUI you want to create.

Below is an example when creating a "luxury" GUI for air conditioner. You have to consider the following:

- Basic requirements
  - Landscape screen
  - Touch screen function

- Design that gives a sense of luxury
  - "Red" color to show three times the performance than other products
  - Add "Gold" color
  - Avoid using flat designs

- Use multiple screens
  - Easier to operate by not putting all functions on one screen.

- Basic display on all screens: Operation status and switching button
  - You can simply "copy & paste" to show on all screens.

- Colorful design
  - Easy to recognize difference functions.

6.2.2 Create an overall image

Use "Search" to get different red color images.

For example, try searching for keywords such as "red, background, commercial free".

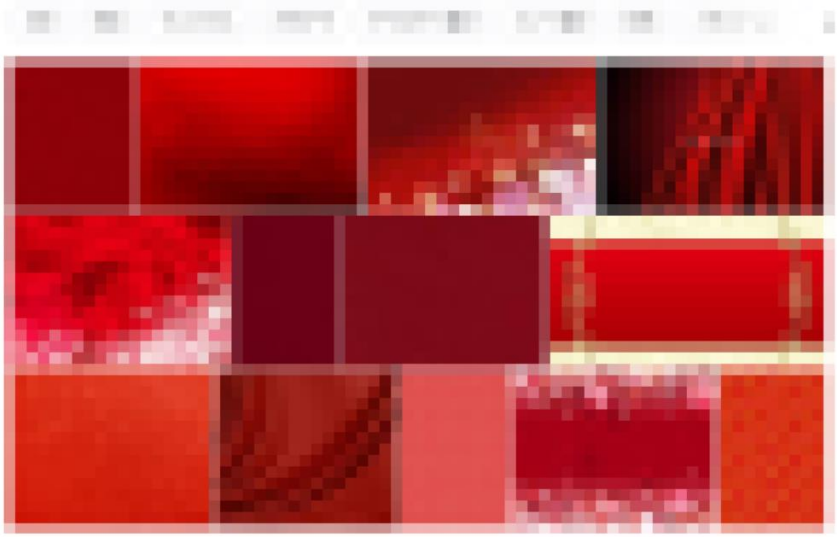


Fig. 111 Search result page (example)

Find your best image and paste it on the slide. If the graphic is for commercial use or open to public, please check the rights or pay the IP owner if needed.

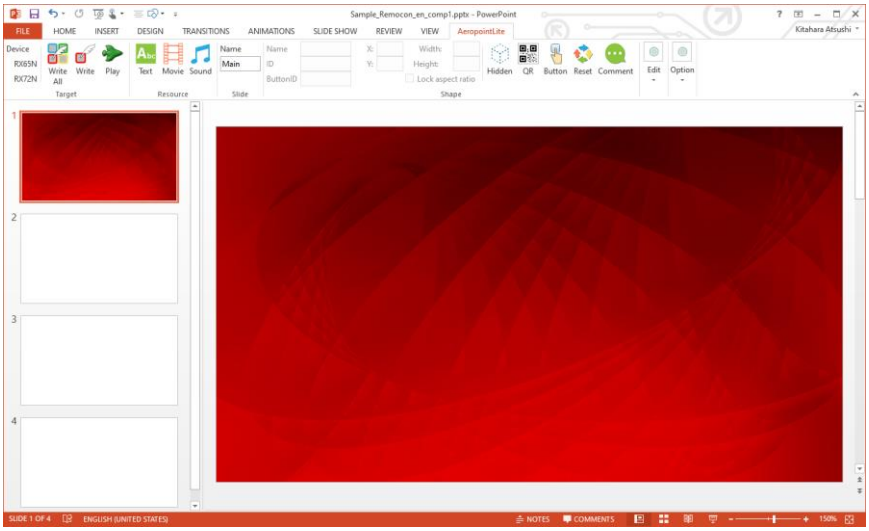


Fig. 112 Paste a picture as background

Preview this onto your actual device to check the color.

You may use WHITE font or near color that matches to the dark background to read the text easily.

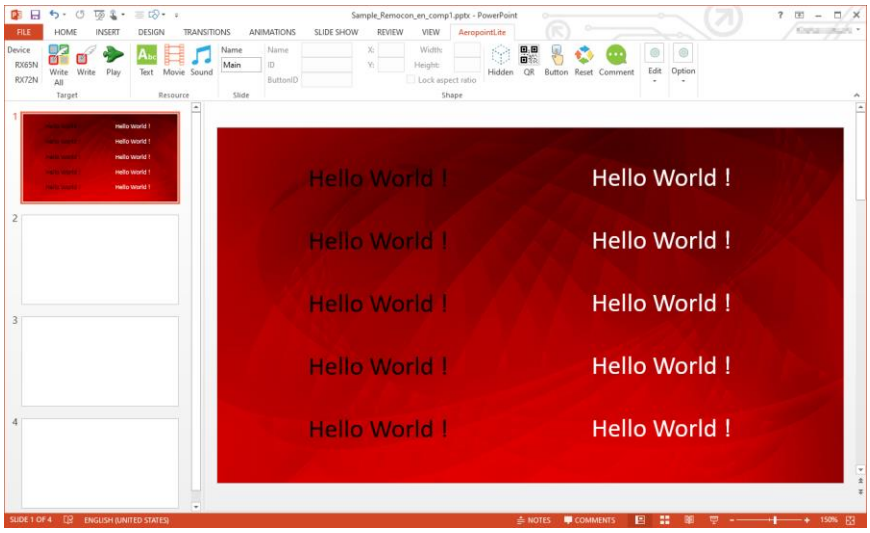


Fig. 113 Consider the color of letters

Place temporary parts of your design to check the overall layout. Place "Status" that is commonly displayed on multiple slides at the top of the screen.

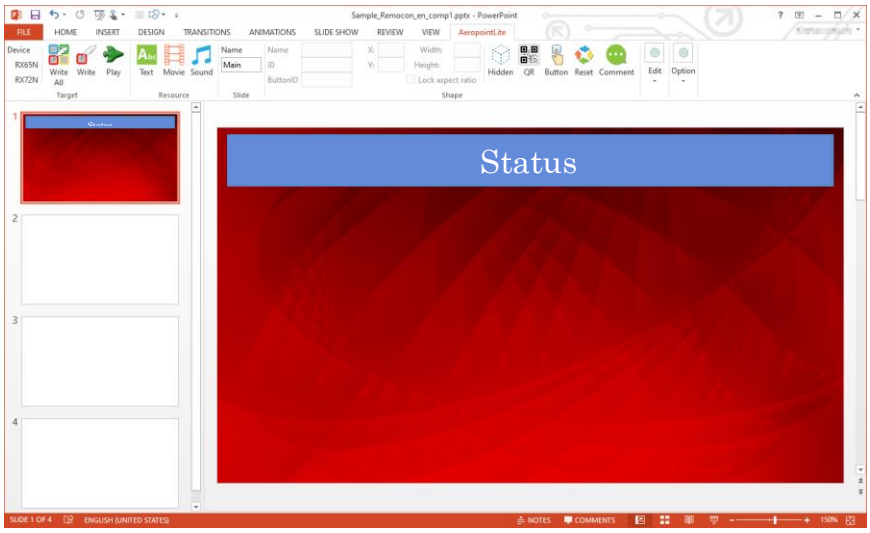


Fig. 114 An example for how to place "Status"

Then place the "operating" functions such as "heating" and "Cooling". This is a rather important but not frequently used function, thus in this sample, we placed on the left side and emphasized each function by different color.

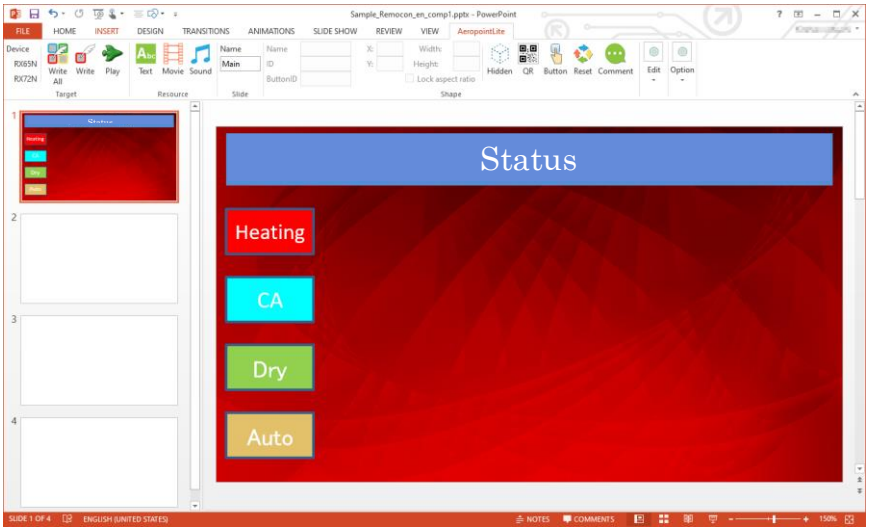


Fig. 115 Place "operation" buttons

Then place LEFT and RIGHT arrows (to go forward or backward) and dots indicators (to show the location of your slide) at the bottom of the screen to switch among the pages.

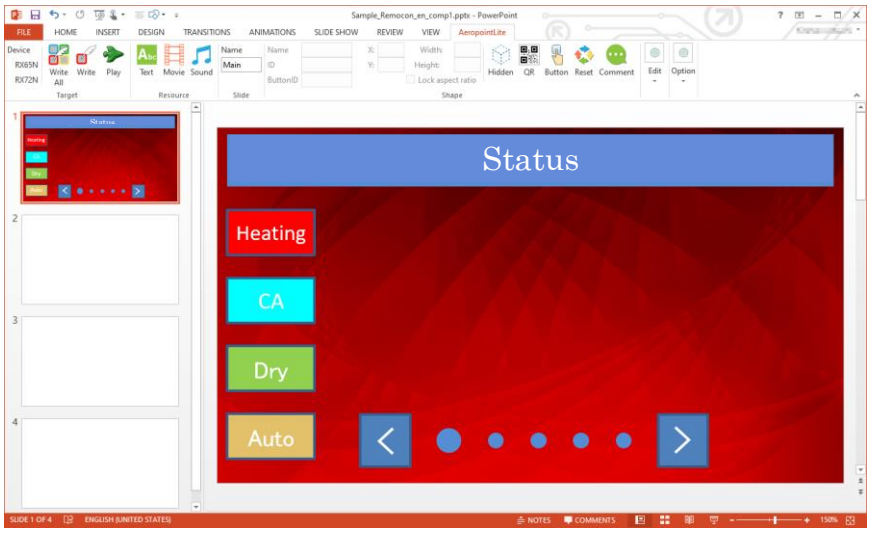


Fig. 116 Place the page switch button

The dots above are for temporary use. Adjust the number of slides afterwards.

First, create a slide to set the main temperature.

The buttons for switching the temperature are most commonly used, so place them on the right-hand side (assuming user as a right-handed person). Two buttons are required and should be placed close together to raise/lower the temperature. The buttons should be big to avoid accidental operation.

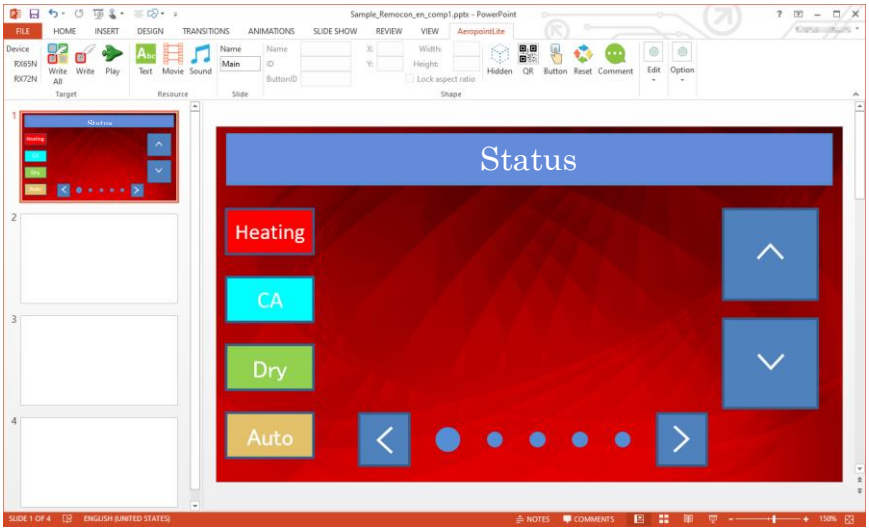


Fig. 117 Place buttons to raise or lower the temperature

Place the most important part ("Temperature" and Degree) in the center of the slide. You may set the decimal point smaller for better presentation.

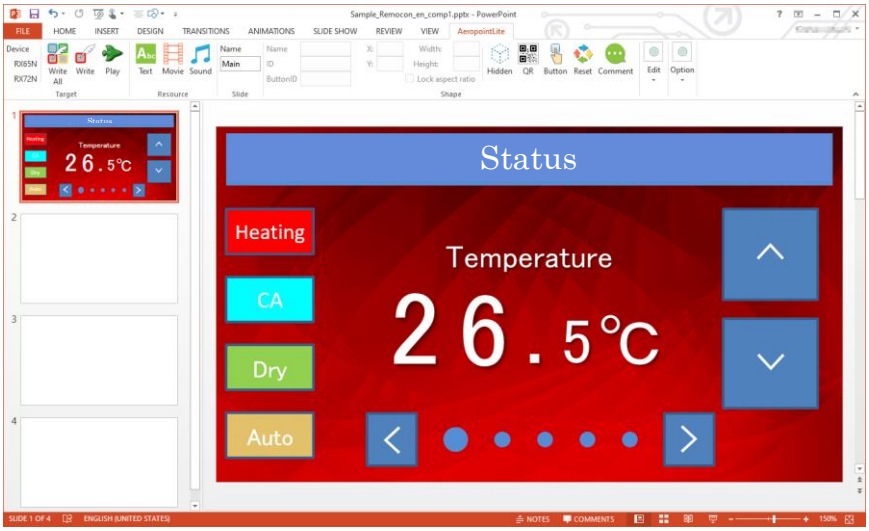


Fig. 118 Place the setting temperature

For GUI design, it is important to shape and arrange the parts in consideration of the function to be used on a touch screen. So far, the draft of the main parts of the GUI has been completed.

By using Aeropoint, you can preview on the actual device at any time whenever your design is done/updated - to check on the screen of the actual device to change the size of the button or character even at early stages of development.

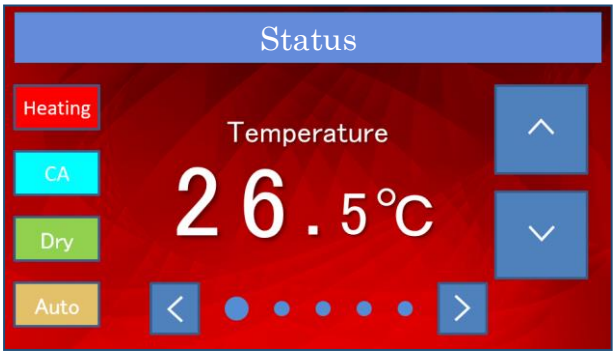


Fig. 119 Display on actual device

Also you can check directly on the actual device to confirm the touch screen panel functions – to check if the buttons are easy to touch or operate correctly. If not, adjust on Aeropoint and check again on the actual device.

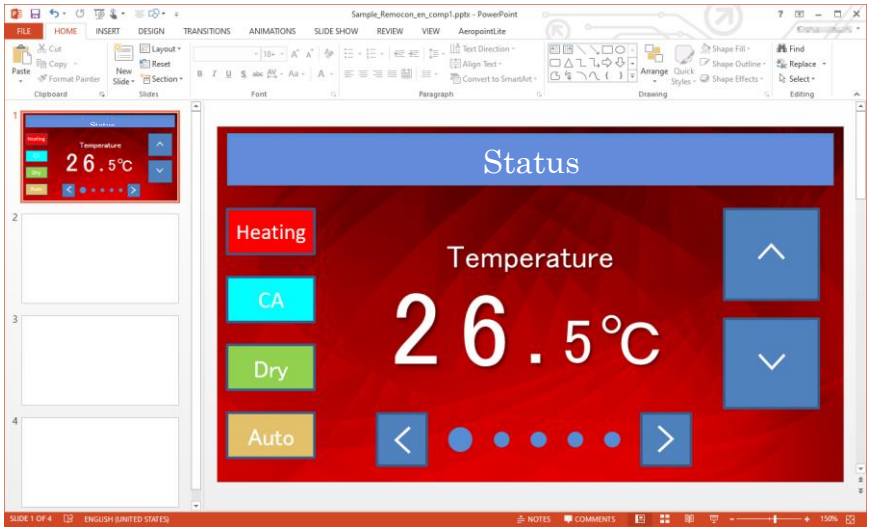


Fig. 120 Fine tuning

6.3 Adjustment

The parts for function placement draft are done. In this part, we will show you how to create with a sense of luxury.

6.3.1 Place the base parts

For "Status" and "Switching" that are commonly used on multiple pages, we use BLACK to show the base parts to separate from the background.

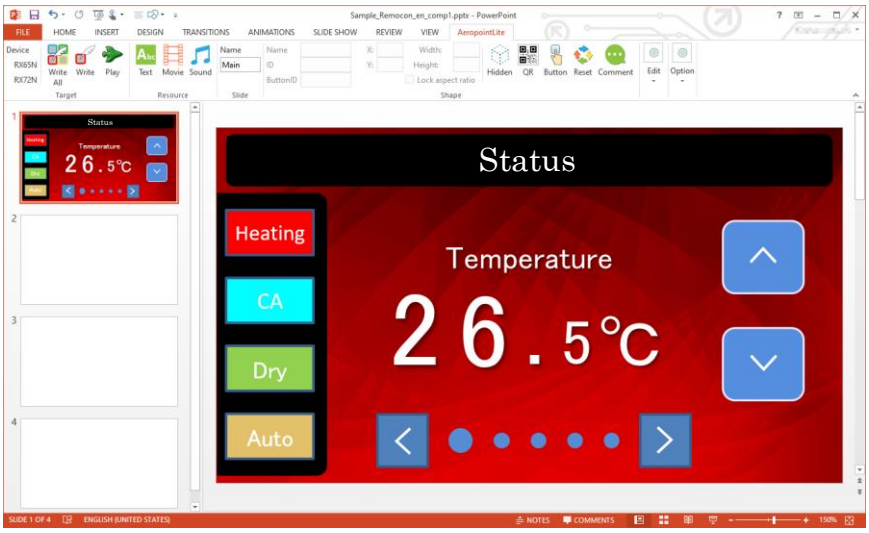


Fig. 121 Place the base parts

After that, preview on the actual device and adjust the size and position.

6.3.2 Making your original buttons for GUI

Your own designs can make the entire GUI unique.
Here we show some examples for the most commonly used buttons for raising or lowering temperature control.

Arrow button

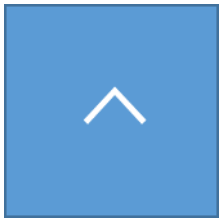


Fig. 122 "Arrow" button

Create some rounded button shapes (like below) and choose the one that fits your design. Here we chose number 2 shape.

Fig. 123 Compare the shapes

Select the best color image (red in this sample) below. Here we chose number 1 color.

Fig. 124 Compare the colors

Then place these designs and preview again.

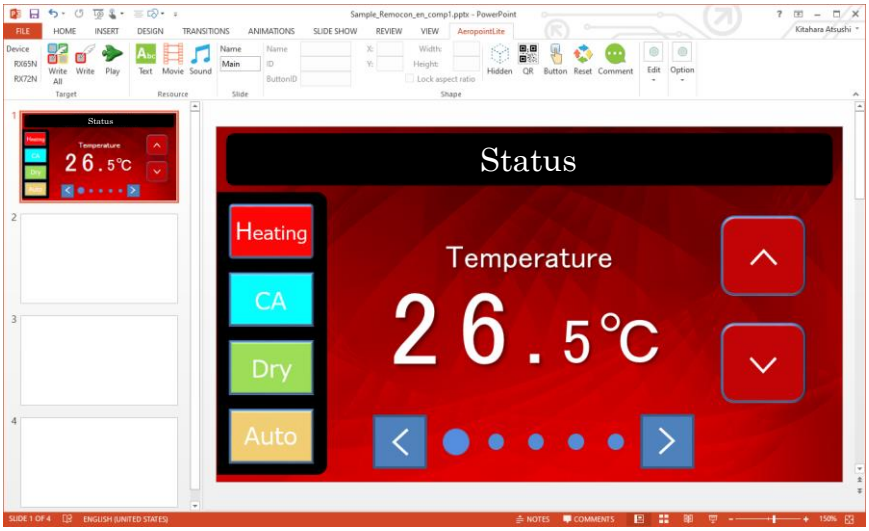


Fig. 125 Place these designs

The shapes are hard to see as the red color is similar to the background color.
Therefore, add gold color to the buttons to improve the visibility and enhance the sense of luxury.

Gold color is a collection of brown, yellow and whitish colors. Search again for your best color.
For example, search for "GOLD" on the site "Picular".

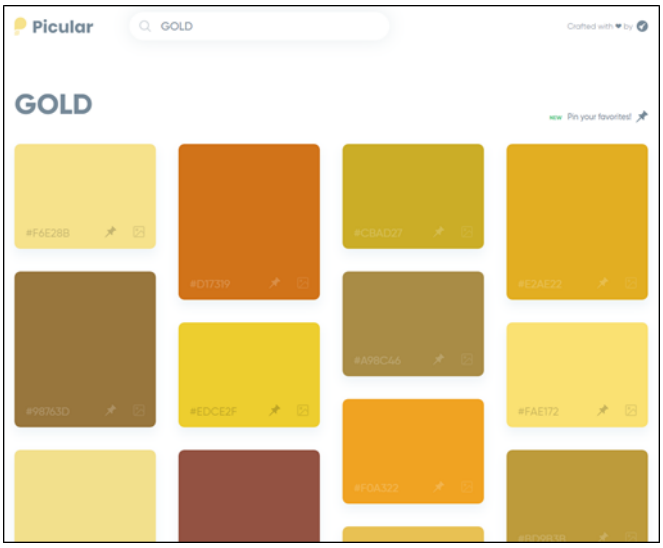


Fig. 126 Search color on "Picular"

In this sample, we use gold as the button edge. Set the edge color by choosing to the color searched above.

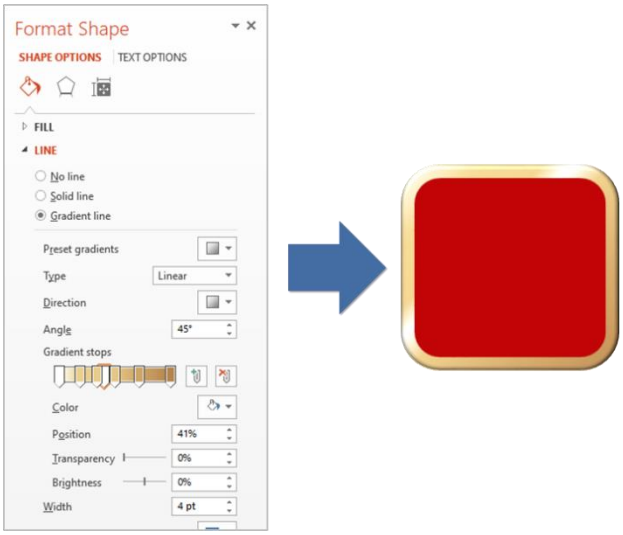


Fig. 127 Gold edge

Place it and preview again.

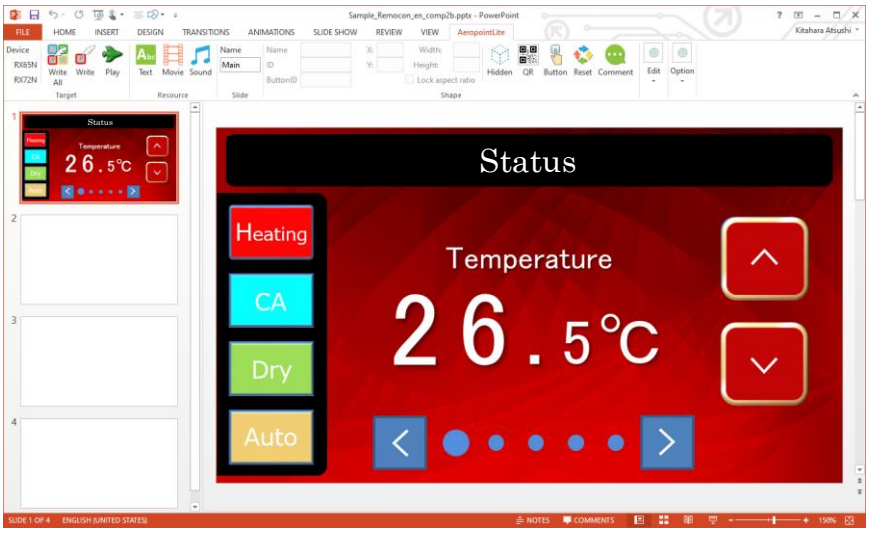


Fig. 128 Place to preview

As the buttons act as an important function for this GUI, you should spend more time to brush up for a better design.

Try to add a more three-dimensional effect to the shape.

Copy this button shape, remove the edges and add a 3D effect to the shape's 3D format.

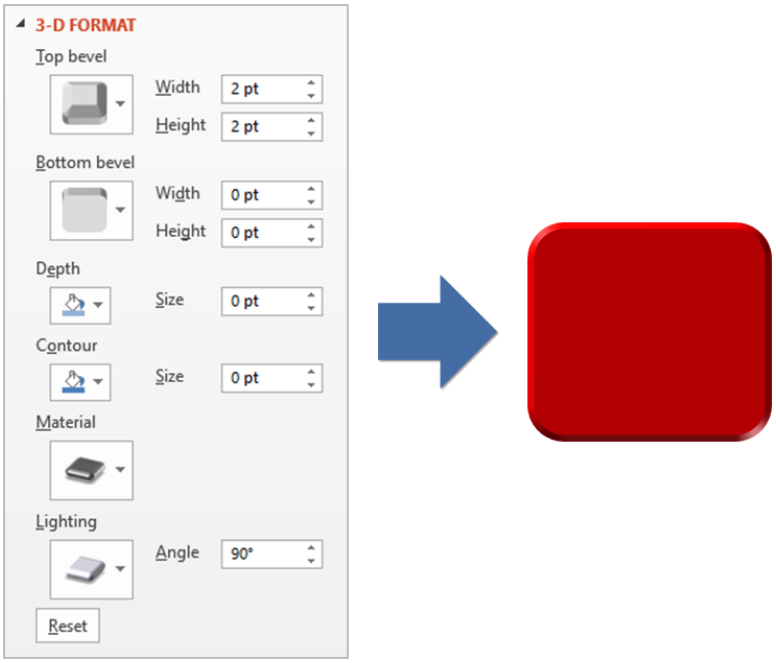


Fig. 129 3D format

Overlap this shape with the original shape.

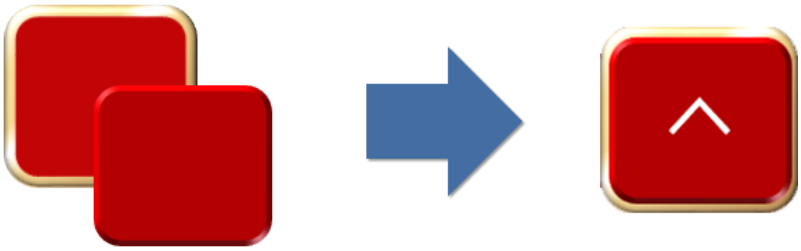


Fig. 130 Overlapping shapes

After the button design, brush up the "arrow" design.

Use two rectangles to create a shape instead of using the preset arrow.

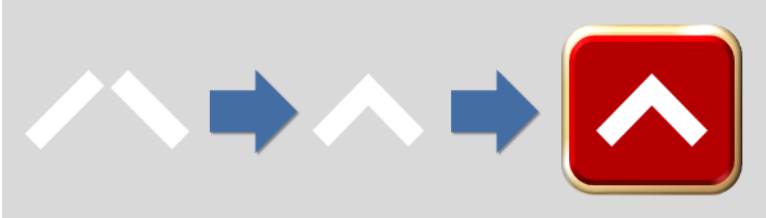


Fig. 131 Create your own arrows

When making the lower button, if you rotate the upper button as it is, the reflection position of the gold edge will also be reversed. Thus, reverse only the arrow inside.

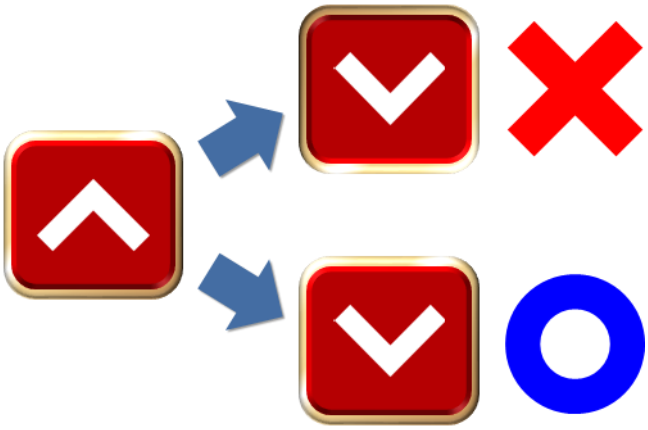


Fig. 132 "Down" Arrow design

Place to preview again.

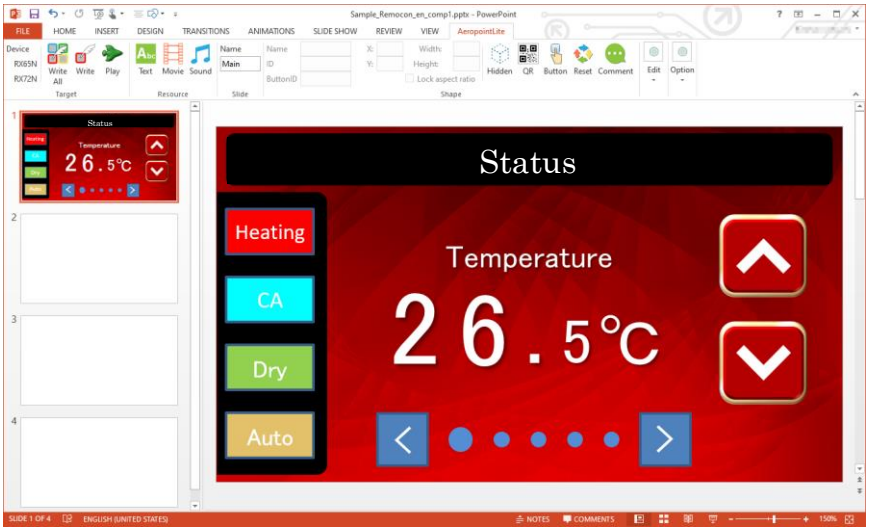


Fig. 133 Place to preview again

6.3.3 Create by copy and paste

As the core button design is done, you can now create the page switching buttons simply by using copy and paste, change the direction of the arrows, and resize.

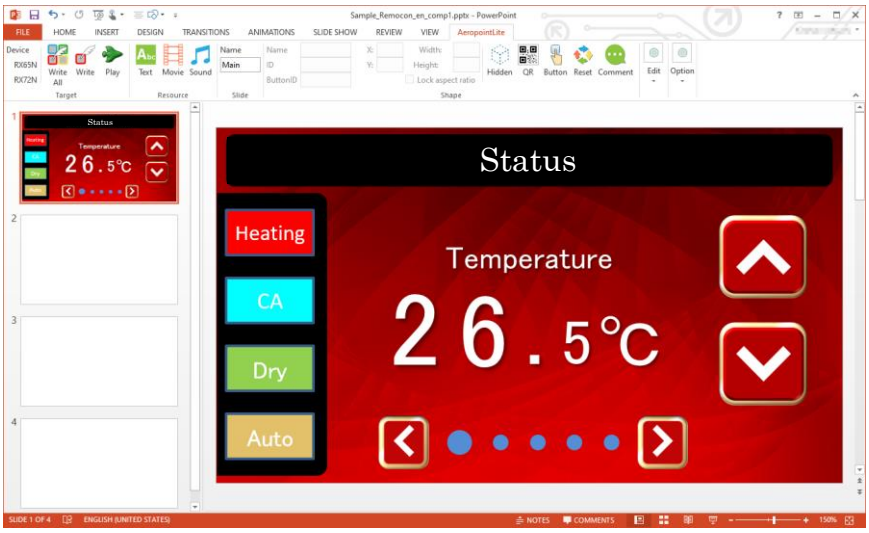


Fig.134 Sample of page switching buttons

However, as these buttons have different functions, and all arrows are squared, we may change the square shape to round shape. Select the button shape and change it to a circle with the PowerPoint's "File" -> "Edit Shape" -> "Change Shape" function.

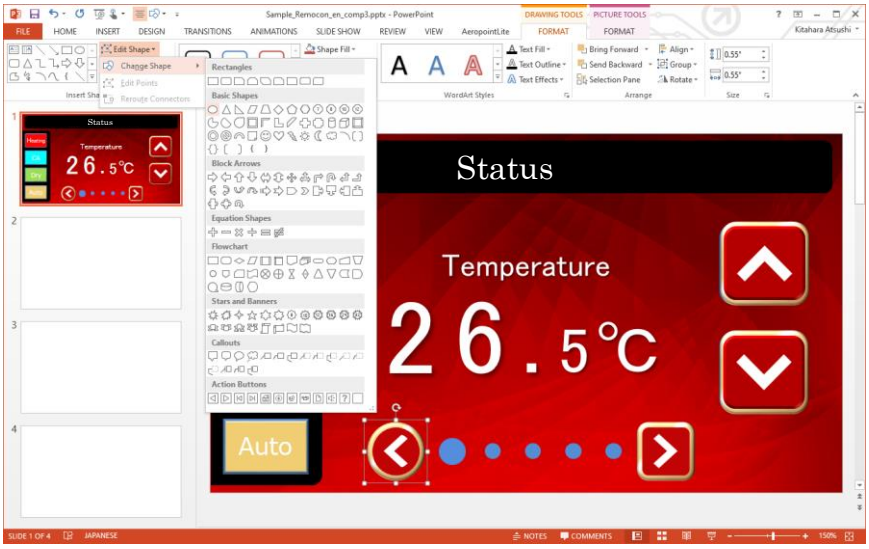


Fig. 135 Change the shape of the button

Next, let's brush up the small circles. These small circles represent the position of the displayed page, and are created by copying the shape of the second layer of this button. When the slide changes, particular small circle representing the page will change the color with reference to the background color.

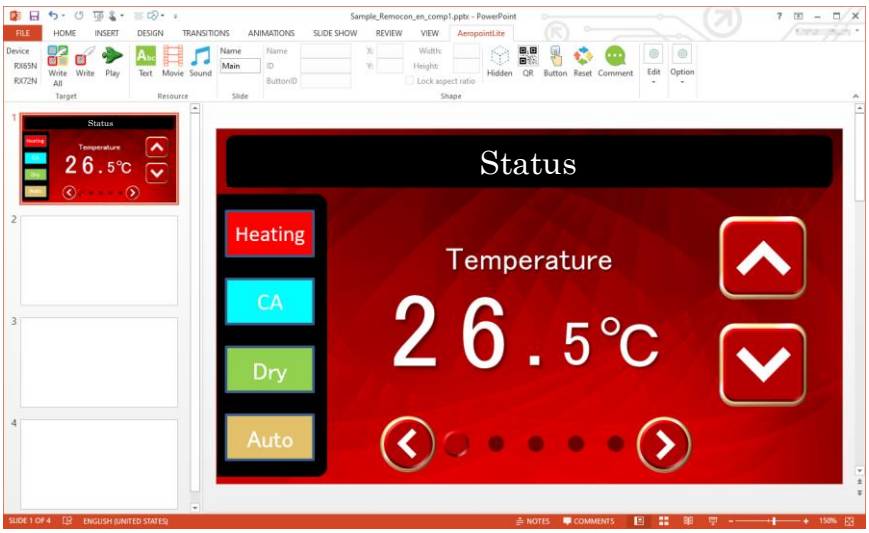


Fig. 136 Button that represents the relative page

Then create the "switching" buttons. You can round the rectangles by copying the arrow shape, resize and change the colors.

Fig. 137 Samples of "switching" button

Adjust the colors.

Fig. 138 Adjust the colors

Place the "switching" button to preview.

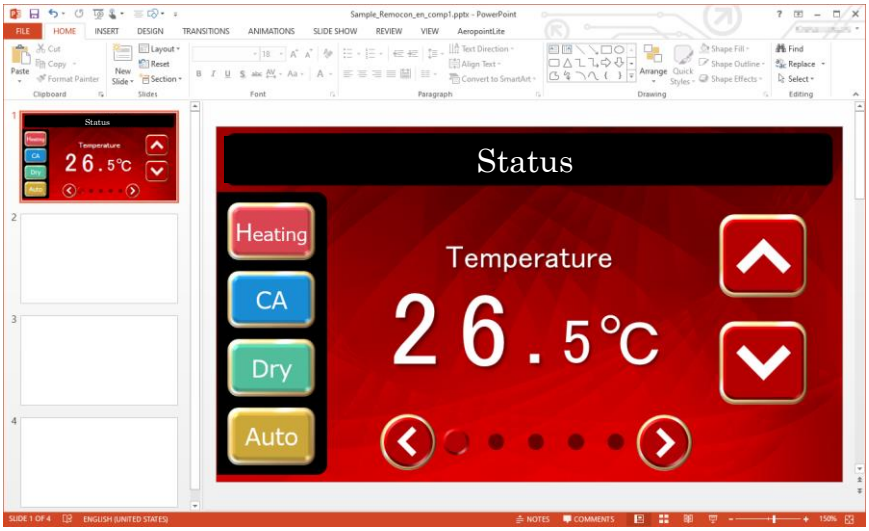


Fig. 139 Place the "switching" button to preview

Next, create "Status". With reference to the conventional model, arrange and add the contents to be displayed. Use a colored icon to make it colorful.

Fig. 140 Create "Status"

Now, all the necessary parts are shown on the slide. If you have time, you may brush up or change the designs.

Then, adjust the center part ("temperature") of the display by changing the font to a circular shape.

Fig. 141 Adjust the center of the display

Preview this design on the actual device.

Fig. 142 Preview and check on the actual device

The above sample is a GUI design draft that gives the sense of luxury feeling. You may refer to this when creating your own unique design.

6.4 Control settings

By using the Aeropoint function, you can simply repeat the process by checking with the actual device to correct/update your design to make it perfect.

6.4.1 Buttons

Using the "button" function of the Aeropoint toolkit, assign button IDs to all the buttons created in the previous section.

Fig. 143 Assign a button ID

Press "Play" on the Aeropoint toolkit to run on the actual device. The "connector" logs such as the button being pressed will be displayed. In actual product development, you create the hardware to respond to this command.

```
[14:09:20]>Button press : Slide 0, Button 0
[14:09:21]>Button press : Slide 0, Button 1
[14:09:21]>Button press : Slide 0, Button 2
[14:09:22]>Button press : Slide 0, Button 3
[14:09:22]>Button press : Slide 0, Button 4
[14:09:23]>Button press : Slide 0, Button 5
[14:09:24]>Button press : Slide 0, Button 6
[14:09:25]>Button press : Slide 0, Button 7
```

Fig. 144 "Connector" logs

You can also create a demo that turns ON the LED on the D-Amp board by setting the "Control output" of the Button Dialog.

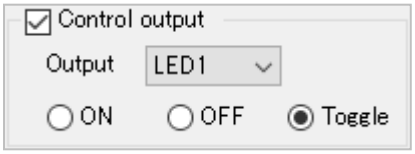


Fig. 145 Control output dialog

6.4.2 Switching screens

If you have 5 slides, copy and paste the first slide for 5 times. Use a bigger circle to show your current page, and place the relative circles that indicate the position as below.

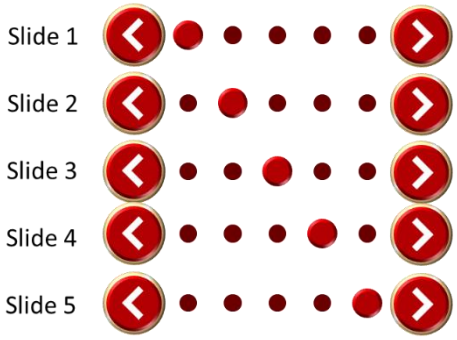


Fig. 146 Page indicators that show your current position

Place the above page indicators to the relative slide like this.

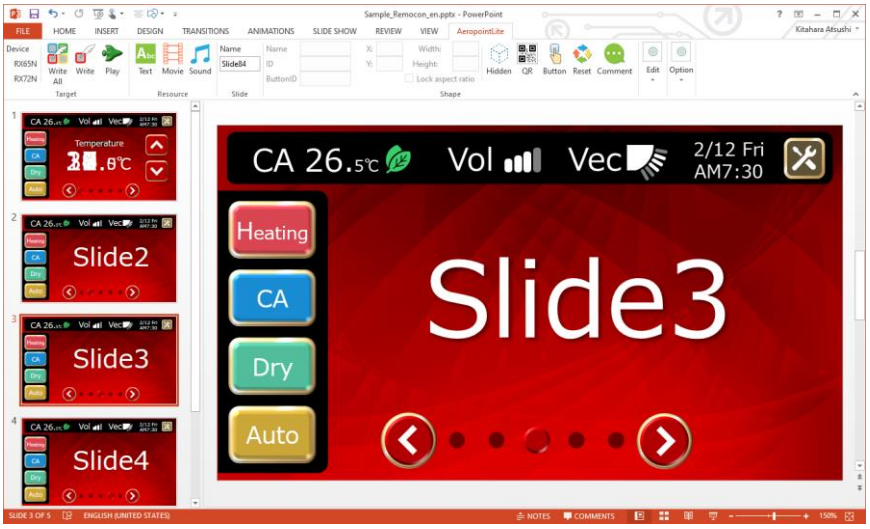


Fig. 147 The page you are viewing is Slide 3

You can use the "Button" function of Aeropoint toolkit, to set to transit to the previous/next page by pressing the left/right button.

Fig. 148 "Button" Dialog

Run on the actual device to test if the left/right button works.

Fig. 149 Switching among the slides

6.4.3 Number

Maximum text size is 60 pixels. You can use the shape display ON/OFF to express changes in numbers. You make as many shapes as you want to change and assign IDs. As an example of ID assignment, "100,200,300" is assigned to the tens digit "1 to 3", "10 to 90" is assigned to the ones digit "0 to 9", and "0,5" We assigned "0,5" to ".

Fig. 150 Assign ID

If you want the shapes to be placed at exactly the same coordinates, it is convenient to enter the X and Y coordinate values directly in the shape group of the Aeropoint toolkit.

When the actual device is displayed at this stage, all number shapes are displayed.

Fig. 151 Display on actual device

Use the Connector's "Shape" function (ON/OFF command) to delete the numbers that you no longer use.

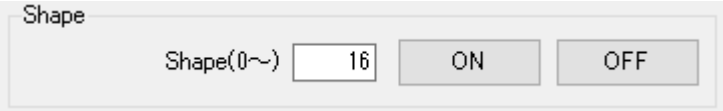


Fig. 152 shape function

Now, the target number is displayed.

Fig. 153 Display on actual device

In this demo, as the linkage between shapes has not yet been set, so you cannot change the set temperature even if you press the upper and lower buttons.

It is necessary to set the command via UART or the program which uses the API of the SDK.
When creating a program, refer to the symbol information output (Chapter "5.6.2 Export").

6.5 Commercialize your prototype

6.5.1 Fine tuning

Though you have checked on the actual device several times, but fine tuning is needed to notice minor mistakes. You may review it some days later and make fine tuning.

6.5.2 Check by someone else

Find 2 or more persons to check the prototype for further advices/comments.

6.5.3 Commercialization

Next step is to commercialize your prototype.

If you want to proceed with verification using the Envision Kit as it is, refer to Chapter "7 SDK"

If you want to develop a command-based product, CRI will customize the command implementation to suit your hardware. Contact CRI(refer to Chapter "14 Contact us").

7 SDK

In the reference SDK, you can use Aeropoint programming with RX72N Envision Kit.

Fig. 155 RX72N Envision Kit

7.1 Application development environment

Prepare the development environment to create the Aeropoint application.

The development environment may be updated by the supplier that are required, e.g. the toolkit, integrated environment, compiler, etc. If you want to use another version other than the one described in this document, proceed with the documentation provided by that supplier.

If you just want to understand the mechanism of Aeropoint, the specifications of API and the programming method, etc., you may skip this configuration the development environment part.

7.1.1 "Renesas Flash Programmer"

Install "Renesas Flash Programmer" on your PC to write the execution binary of the sample program directly to the microcomputer.

- [Free-of-charge edition] Renesas Flash Programmer V3.06.01
  - https://www.renesas.com/software/D4001218.html

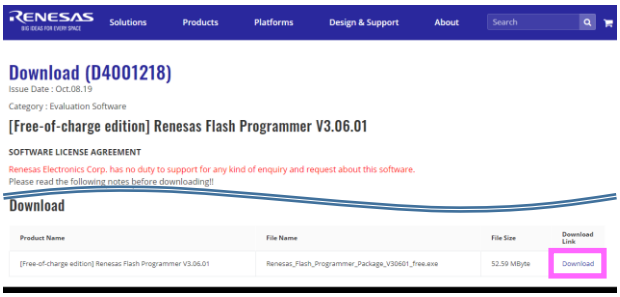


Fig. 156 Download page

7.1.2 Aeropoint Toolkit

You need Aeropoint toolkit to add-in to PowerPoint®.
Install either of the following toolkits:

- Aeropoint Toolkit
  - 32-bit version : AeropointLiteToolkit_xxxxxxxxRx_CRI_32bit.exe
  - 64-bit version : AeropointLiteToolkit_xxxxxxxxRx_CRI_64bit.exe

7.1.3 RX72N Envision Kit

Prepare the RX72N Envision Kit from Renesas Electronics Corporation (Hereafter referred to as "Renesas").

- RX72N Envision Kit
  - https://www.renesas.com/products/software-tools/boards-and-kits/eval-kits/rx72n-envision-kit.html

7.1.4 e²studio

An integrated environment for programming. Download the e² studio installer from the Renesas website and follow the instructions to set up.

- e² studio V7.8.0 installer (Offline installer)
  - https://www.renesas.com/software/D4001318.html

- e² studio 2021-01 installer (Offline installer) < FIT kit compatible version
  - https://www.renesas.com/document/uid/e-studio-2021-01-installer-offline-installer

The following is the procedure for setting up
>If your PC shows windows protected, then Run anyway

1. Welcome page
   - Click "Next"
2. Installation folder
   - Decide where to install (or use default), and click "Next"
3. Device family
   - Select "RX device support" checkbox, and click "Next"
4. ※(Additional components)
   - Check "Japanese language pack", and click "Next"
5. Components
   - Click "Next"
6. Additional software
   - Check "Renesas CCRX v3.01.00" in the "Toolchains & Utilities" tab, click "Next"
   - If the toolchain is installed, the list will not be displayed.
   - Toolchain can also be installed separately.
     - RX Compiler CC-RX V3.01.00
       - https://www.renesas.com/software/D4000988.html

After the above procedures, follow the Installer's instructions. After the setup is complete, confirm the start of e² studio and set the tool chain.

7.1.5 "e² studio" Toolchain

Start the installed e² studio and set the tool chain.

1. Start "e² studio"
   - Run the installed "e² studio"
2. Eclipse Launcher
   - Specify an arbitrary path for the workspace, and click "Start"
3. Toolchain Integration
   - Select "Renesas CCRX-v3.01.00" checkbox, and click "Register"
   - This window below is displayed only the first time you start the toolchain after installation.

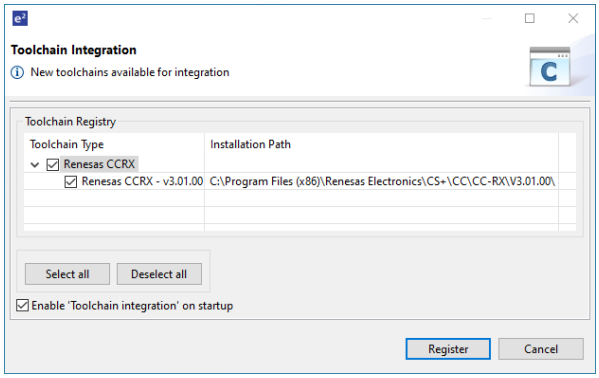


Fig. 157 Toolchain Integration

4. Welcome page of "e² studio"
   - If this screen is displayed, installation and settings are complete.
   - If you miss the setting in "Toolchain Integration", you can set it by the following procedure:
     - Select "Help" → "Add Renesas Toolchain" from the Menu bar
     - Check "v3.01.00" of "Renesas CCRX" and click "Apply and Close"

If you can't find "Renesas CCRX v3.01.00" in "Toolchain Integration" or "Add Renesas Toolchain", you may not have installed "Renesas CCRX v3.01.00".
As described in Step 6 of "7.1.4 e² studio ", the tool chains can be installed individually. (Download to install "RX Compiler CC-RX V3.01.00")

7.2 System configuration

Aeropoint GUI provides Aeropoint Lite Runtime API (hereafter referred to as "Aeropoint runtime") as a library for application development. The Aeropoint player used to play the sample data is also an application developed using this runtime.

Aeropoint Implementation Software (hereafter referred to as "AIS"), which is software for absorbing the differences between microcomputers, is implemented on top of the hardware driver. AIS may implement customized microcomputers and hardware specifications according to the customer's product development process.

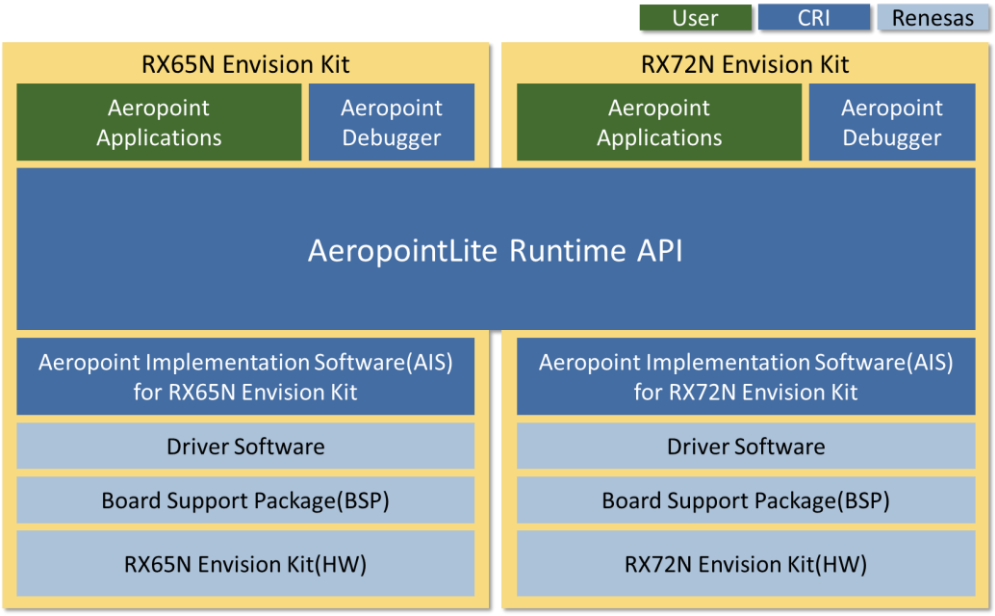


Fig. 158 System configuration

Application development using Aeropoint runtime uses two types of APIs, "Player API" and "Runtime Engine API".

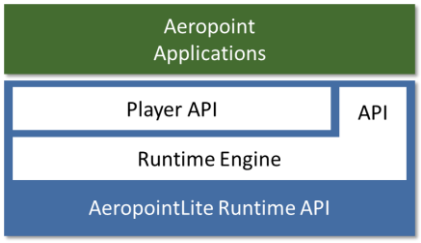


Fig. 159 Runtime API configuration

"Player API" contains the basic programs that are required to control the actual device. So you only have to write limited programming that are for initializations, terminations, and main process calls, etc.

7.3 Basic flow

The following is a basic flow diagram from creating GUI with PowerPoint® until the customer operates the UI on the actual device.

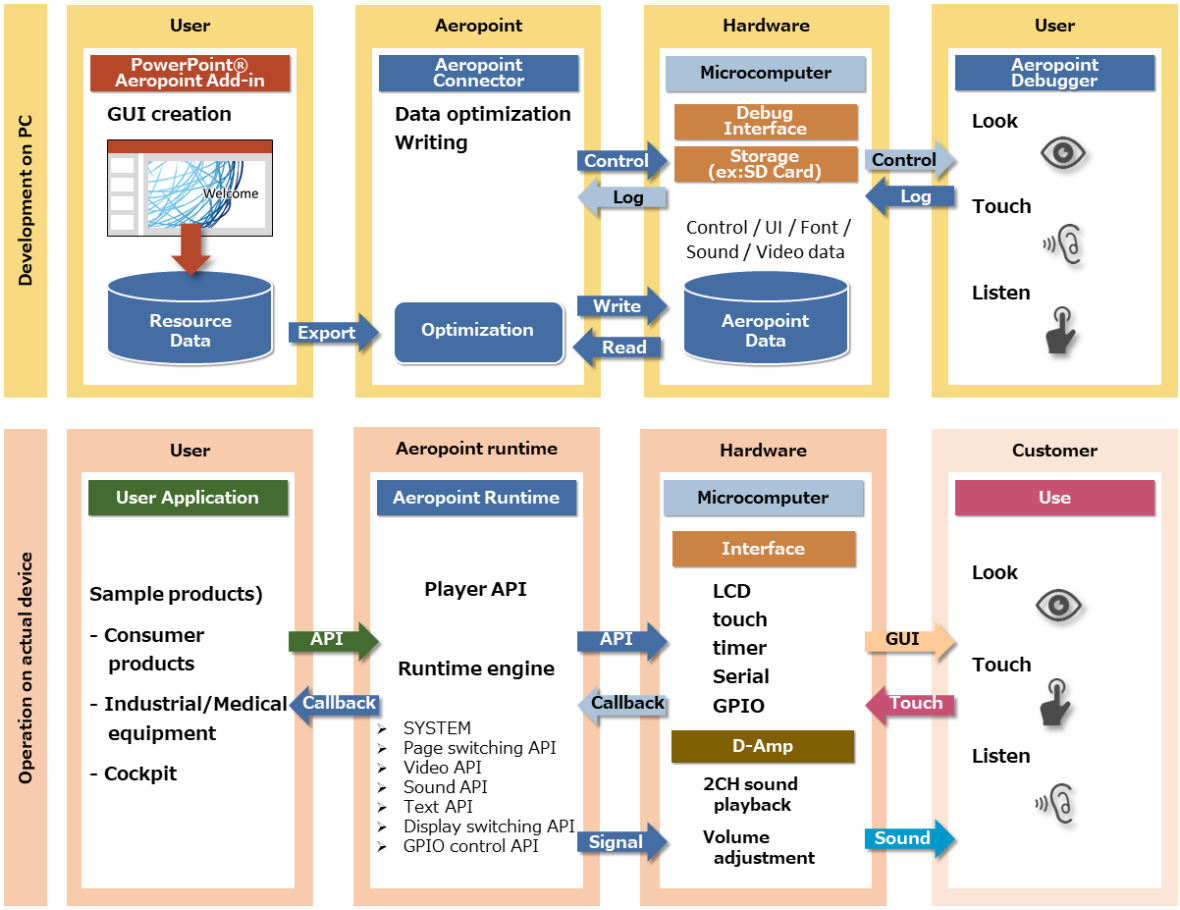


Fig. 160 Basic flow

Develop the UI design and the basic behavior of the application using PowerPoint® on your PC. During the application creation, you can preview the functions on the actual device, to "check, listen, or touch".

You can develop a basic application using the player API. If you want to add features for each application or run complex specifications to the application, use various APIs in the runtime engine for customization.

7.4 Aeropoint runtime

"Runtime Engine" in Aeropoint runtime controls and executes the functions such as UI, sound, and video on the actual device. This "Runtime Engine" performs the basic operation of the content based on the data created using PowerPoint® in this case.

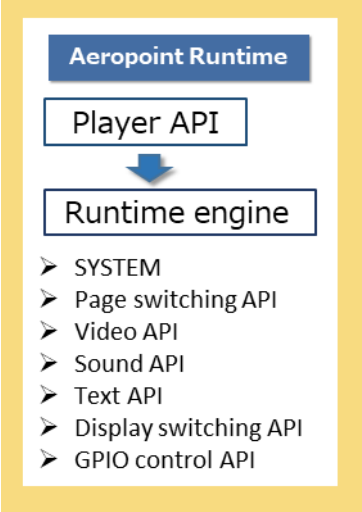


Fig. 161 "Player API" and "Runtime Engine"

Since there are many "runtime engine" APIs, if you simply use them will increase the amount of code required for implementation. Therefore, "Aeropoint Runtime" provides the "Player API" that summarizes the operation of "Runtime Engine" and performs the basic operation of the application. This "Player API" is the basic mechanism for you to create Aeropoint applications.

7.4.1 Minimal application

The minimum application can be completed by the three procedures of "initialization, main processing and termination" of the Player API. The "main process" function continues to call the necessary processing, e.g. UI animation, video display, etc., with unlimited times. This program that repeatedly calls the main processing is called "main loop".

```
1  void main(void)
2  {
3      CriBool flag = CRI_TRUE;
4
5      /* Initialize */
6      CriAeroPlayer_Initialize();
7
8      /* Main loop */
9      while( flag )
10     {
11         /* Main */
12         flag = CriAeroPlayer_Main();
13     }
14
15     /* Finalize */
16     CriAeroPlayer_Finalize();
17
18     return;
19 }
```

Fig. 162 Minimal application

The workflow of this application is as below:

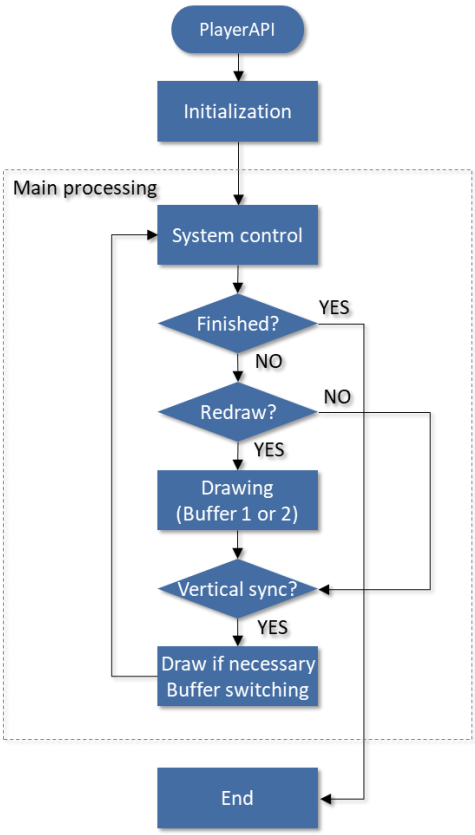


Fig. 163 Minimal application flow

This program is a "view/listen/touch" application that reads data from a storage (such as an SD card), displays a UI, plays a touch interface, and plays sounds.

"Runtime Engine" draws to the screen in the hidden back buffer of the two buffers. "Runtime Engine" waits for the vertical sync signal, which is the screen switching timing, when drawing is completed. Then, it switches between the front buffer that are being displayed, and the back buffer. This operation realizes screen switching without flicker.

Run the smallest application on the actual machine.

1. Preparation of actual device

   Set SW1 / SW3 of "Envision Kit" as shown below.
   Connect the ECN1 terminal beside SW1 and the PC with Micro USB.

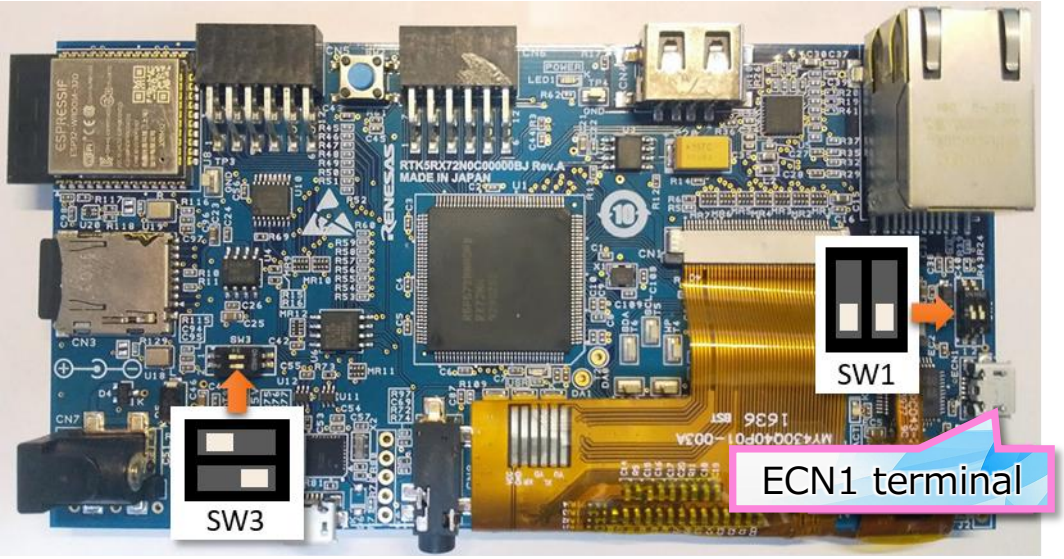


Fig. 164 SW1 / SW3 / ECN1 terminal

2. Launch "e² studio"

   You launch "e² studio".
   The workspace can be any path.
   Please note that an error will occur in the build if the total path length including the projects to be imported later exceeds 260 characters.

3. Start import window

   Right-click on Project Explorer and select Import.

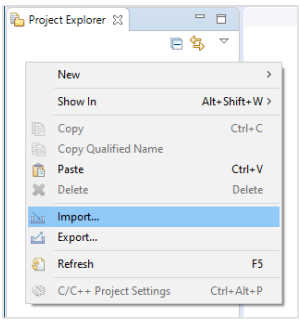


Fig. 165 Select Import

4. Select Import Existing Project

In the import window select "General" > "Existing Projects into Workspace"

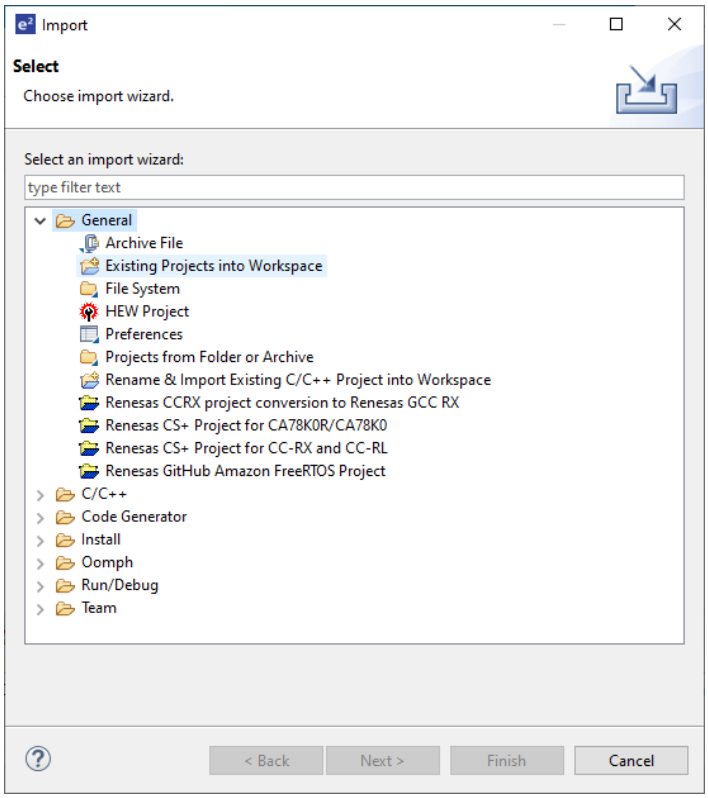


Fig. 166 Select "Existing Project into Workspace"

5. Select ZIP file of library project

Select "Select Archive File", select the ZIP file of the library project in the package, and press "Finish".
You can find the ZIP file of the library project in the following path of the package.

・Runtime/AeropointRX72N_Library.zip

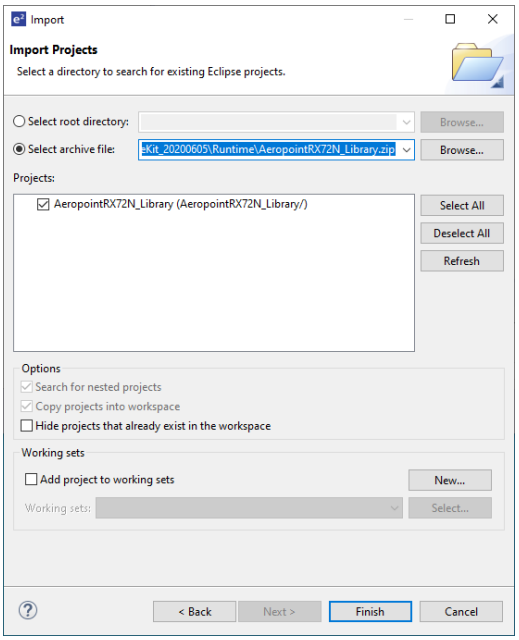


Fig. 167 Select ZIP file of library project

6. Import Player Project

Follow the same procedure as above to import the player project ZIP file.
The player project ZIP file is located in the following path of the package.

・Runtime/AeropointRX72N_Player.zip

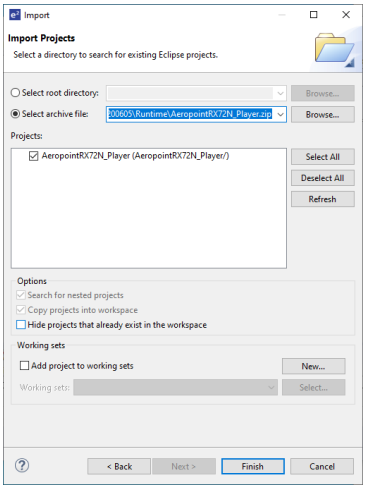


Fig. 168 Select ZIP file of player project

7. Confirm two projects import

Make sure you have two projects in your project explorer: the library project "Aeropoint RX72N_Library" and the player project "Aeropoint RX72N_Player"

8. Build Player Project

Select the player project, right click and select "Build Project".

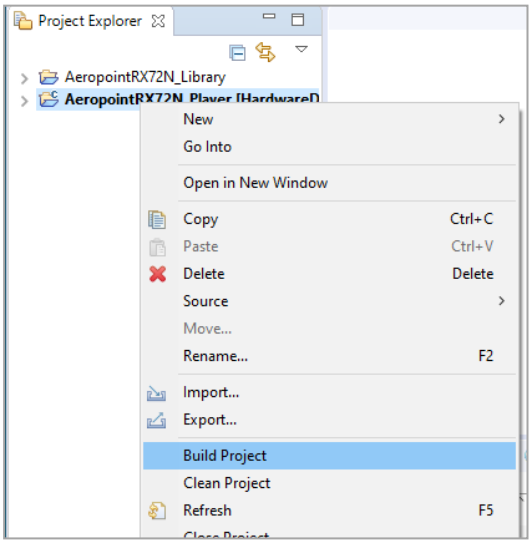


Fig. 169 Build player Project

If the build fails with the error "[ERROR] No toolchain set or toolchain not integrated", the toolchain may not be set. Refer to "7.1.5 "e² studio" Toolchain" to set the toolchain.

9. Check the configuration and launch in "Debug" mode

Confirm that the configuration is "Aeropoint RX72N_Player_Release" and click the Debug execution button. If the configurations are different, select "Aeropoint RX72N_Player_Release" from the down arrow to the right of the configurations.

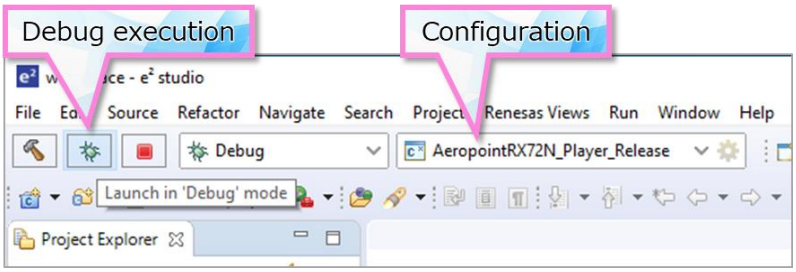


Fig. 170 Configuration and Debug execution button

10. Confirm Perspective Switch

"Perspective switching confirmation" will be displayed during the transition to the debug mode, so select "Yes".

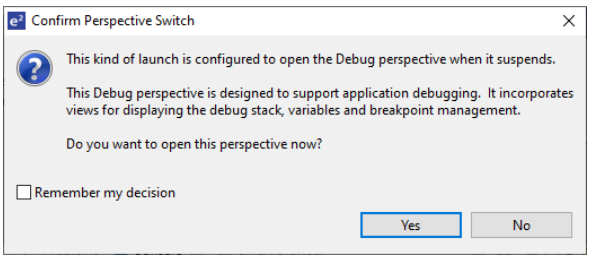


Fig. 171 Confirm Perspective Switch

11. Program execution

When the program is executed, it will be waiting for the first process. Click the "Resume" button twice.

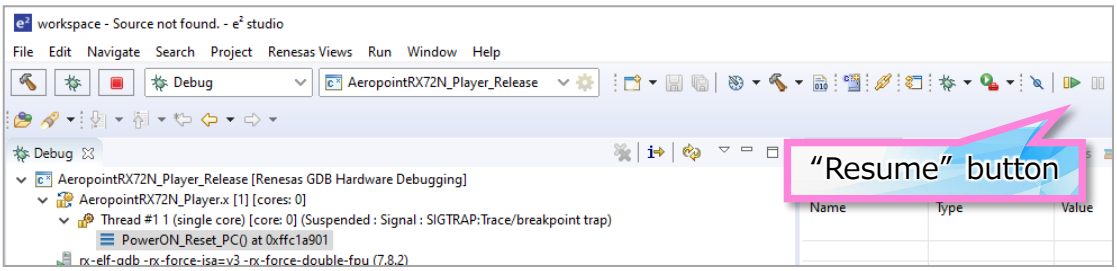


Fig. 172 "Resume" button

12. Display startup screen

The startup screen is displayed on the actual device.

Fig. 173 Startup screen

7.4.2 Display the page

Let's add a program to the smallest application and display the page created with PowerPoint®.

> It is necessary to insert the SD card with the data image written, which is included in Reference kit, into the microSD slot of the actual device. If the data image has not been written to the SD card, prepare according to "11 Write data image"

Select the main program from the "Project Explorer" of "e² studio" and double-click it.

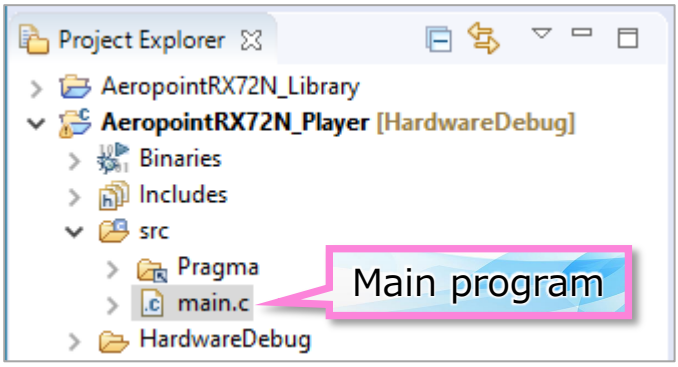


Fig. 174 Select main program

The main program opens in the edit window.

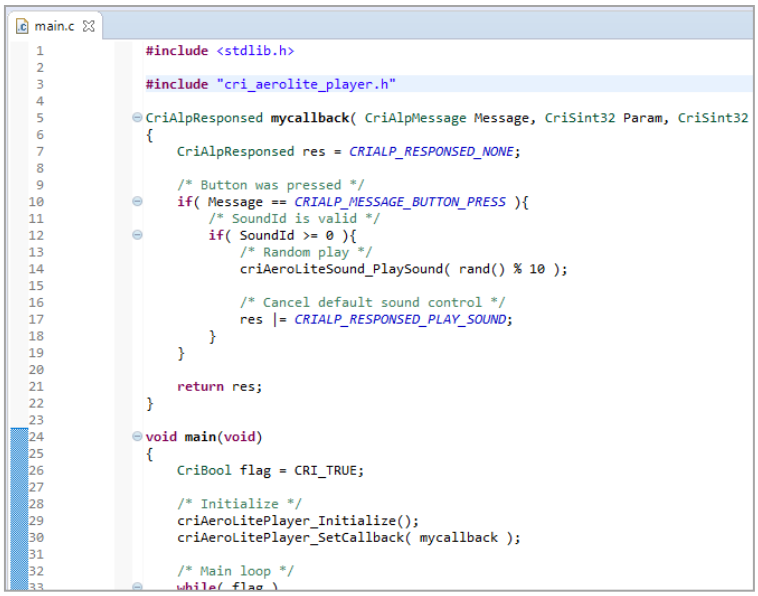


Fig. 175 Main program

Immediately before the main loop starts, add the command "CriAeroPlayer_ReadPage(0)" to display page 0. The part in red letters below.

```
1   void main(void)
2   {
3       CriBool flag = CRI_TRUE;
4
5       /* Initialize */
6       CriAeroPlayer_Initialize();
7
8       /* ReadPage 0 */
9       CriAeroPlayer_ReadPage(0);
10
11      /* Main loop */
12      while( flag )
13      {
14          /* Main */
15          flag = CriAeroPlayer_Main();
16      }
17
18      /* Finalize */
19      CriAeroPlayer_Finalize();
20
21      return;
22  }
```

Display page 0

Fig. 176 Added description to display page 0

You save the program, build and run it again. It is successful if the following "Welcome screen" is displayed.

Fig. 177 Welcome screen

After confirming the operation, restore the modified program.

7.4.3 Callback

The callback functions to inform the user function of the event such as button tap on the actual device.

With the Aeropoint toolkit, you can specify slide changes, sound/video playback, etc., as what happens when you tap a button. In addition, by responding to callbacks, you have the flexibility to extend the behavior of your application, such as changing their behavior or driving your own hardware.

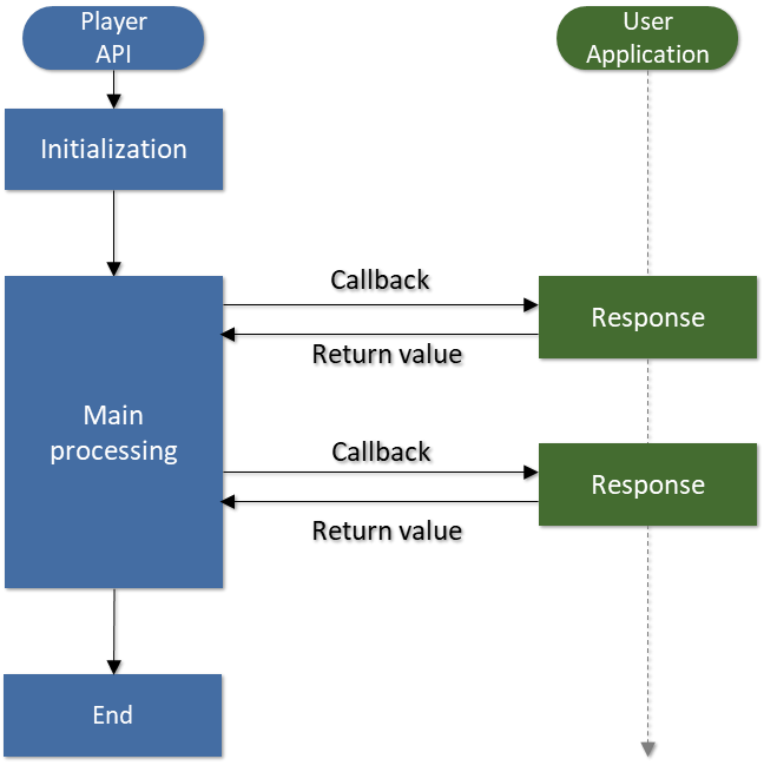


Fig. 178 Callback flow overview

Let's see how it works with a sample callback. Use the following project and GUI data.

- Project
  - ✧ Sample\Project\AeropointRX72N_Sample_Callback.zip
- GUI data
  - ✧ Sample\GUI\Sample_Callback.pptx

You import this project into e² studio. And write the GUI data to the SD card by referring to "5.1.1 "Write All"".

When you open the main.c of e² studio, the callback function is described main.c。 See the reference for more details on using functions. "7.5 Aeropoint programming reference".
The code below cancels the playback behavior of the specified sound and plays a random sound.

```
CriAeroResponsed mycallback( CriAeroMessage Message, CriSint32 Param, CriSint32 SoundId, CriSint32 MovieId )
{
    CriAeroResponsed res = CRI_AERO_RESPONSED_NONE;

    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){
        /* SoundId is valid */
        if( SoundId >= 0 ){
            /* Random play */
            CriAeroSound_PlaySound( rand() % 10 );

            /* Cancel default sound control */
            res |= CRI_AERO_RESPONSED_PLAY_SOUND;
        }
    }

    return res;
}
```

If the button is pressed

Play sound if ID is 0 or more

Random sound ID

Cancel default behavior

Fig. 179 Example with a callback function

By registering the callback function with the Aeropoint runtime before starting the main loop, it will be called when the event occurs.

```
1   void main(void)
2   {
3       CriBool flag = CRI_TRUE;
4
5       /* Initialize */
6       CriAeroPlayer_Initialize();
7
8       /* Set UserCallback */
9       CriAeroPlayer_SetCallback(mycallback);
10
11      /* Main loop */
12      while( flag )
13      {
14          /* Main */
15          flag = CriAeroPlayer_Main();
16      }
17
18      /* Finalize */
19      CriAeroPlayer_Finalize();
20
21      return;
22  }
```

Set callback

Fig. 180 Set callback

If you build and run this program, you'll see the startup screen because it doesn't have the code to display page 0 added by the minimal program. When you press "play" button of Aeropoint add-in or the actual SW2 switch, the screen of "Callback Test" is displayed.

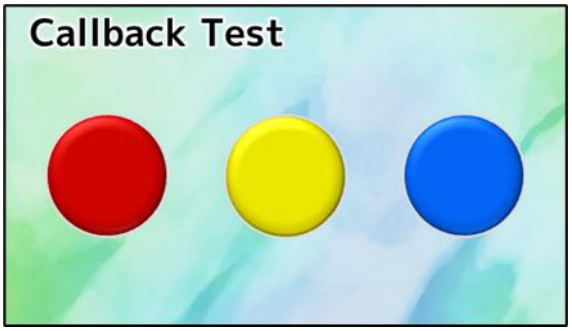


Fig. 181 "Callback Test" screen

When you press the button, the sound set in the Aeropoint add-in should sound, but the callback function works and the sound sounds randomly.

7.5 Aeropoint programming reference

We will explain the symbols, structures, functions, etc. required to create Aeropoint applications.

7.5.1 Symbol, structure, enum

Color specification enum. Used to specify colors such as fonts.

```
/* Color specification */
enum {
        CRI_AERO_COLOR_PUNCHTHROUGH = 0,    /* Punch through (background color) */
        CRI_AERO_COLOR_RESERVED_1,          /* RESERVED_1          */
        CRI_AERO_COLOR_RESERVED_2,          /* RESERVED_2          */
        CRI_AERO_COLOR_BLACK,               /* BLACK               */
        CRI_AERO_COLOR_WHITE,               /* WHITE               */
        CRI_AERO_COLOR_RED,                 /* RED                 */
        CRI_AERO_COLOR_GREEN,               /* GREEN               */
        CRI_AERO_COLOR_BLUE,                /* BLUE                */
        CRI_AERO_COLOR_YELLOW,              /* YELLOW              */
        CRI_AERO_COLOR_CYAN,                /* CYAN                */
        CRI_AERO_COLOR_MAGENTA,             /* MAGENTA             */

        CRI_AERO_COLOR_END
};

typedef CriUint32 CriAeroColorID;
```

LED specification enum for testing. It is used to specify the LED.

```
/* Test IO LED color specification */
enum
{
        CRI_AERO_TESTIO_LED_GREEN = 0,
        CRI_AERO_TESTIO_LED_BLUE,
        CRI_AERO_TESTIO_LED_YELLOW,
        CRI_AERO_TESTIO_LED_RED,
        CRI_AERO_TESTIO_LED_WHITE,

        CRI_AERO_TESTIO_LED_NUM
};

typedef CriUint32 CriAeroTestIOLedColor;
```

7.5.2 UART Command

List of UART commands.

Separate each command and parameter with a space. The command ends with ＼n.

Please refer to "8 How to connect to other equipment" about the connection method of the serial communication with UART.

Content	Command	Parameter
Display the specified slide	slideno <NO>	NO：Slide number (1~)
Display the specified shape	shapeon <ID>	ID：Shape ID(0~)
Hide the specified shape	shapeoff <ID>	ID：Shape ID(0~)
Display text	textpring <ID> <X> <Y> <fontID> <ColorID> <string>	ID：Text ID(0~7) X, Y：Coordinates to display fontID：Font ID(0～15) ColorID：Letter color(0～7) String：Any string(UTF-8)
Erase all text	textclear	-
Show title	showtitle	-
Play sound	soundplay <ID>	ID：Sound ID(0~31)
Stop sound	soundstop	-
Change BGM volume	bgmvolume <Vol>	BGM volume (0 to 255)
Change SE volume	sevolume <Vol>	SE volume (0 to 255)
Play video	movieplay <ID>	ID：movie ID(0~7)
Stop video	moviestop	-
Output log	log <string>	string：Arbitrary character string (UTF-8)
Screen redraw	scrupdate	-

7.5.3 Player API

The Player API provides the basic mechanism used by all user applications.

Include the following header file.

```
#include "cri_aero_player.h"
```

List of functions.

Function	Content
CriAeroPlayer_Initialize()	Initialize the player
CriAeroPlayer_Main()	Main process that is called repeatedly
CriAeroPlayer_Finalize()	Exit the player
CriAeroPlayer_Callback()	Callback function
CriAeroPlayer_SetCallback()	Registration and cancellation of callback functions

CriAeroPlayer_Initialize()

Overview

Initialize the player

Format

void CRIAPI CriAeroPlayer_Initialize();

Argument

None

Return value

None

Explanation

Initialize the player and launch the Aeropoint runtime system.

Example of use

```
void main(void)
{
    CriBool flag = CRI_TRUE;

    /* Initialize */
    CriAeroPlayer_Initialize();

    /* Main loop */
    while( flag )
    {
        /* Main */
        flag = CriAeroPlayer_Main();
    }

    /* Finalize */
    CriAeroPlayer_Finalize()

    return;
}
```

Note

Recalling will return without doing anything.

CriAeroPlayer_Main()

Overview

Main process that is called repeatedly

Format

CriBool CRIAPI CriAeroPlayer_Main();

Argument

None

Return value

CRI_TRUE	:	Continue processing
CRI_FALSE	:	Processing Exit

Explanation

It displays the UI, plays sounds and videos, controls touches and GPIOs, and performs all the necessary processing for Aeropoint to work. You need to keep calling it regularly.
If the return value is "CRI_FALSE", you exit the main loop and call the player's exit function.
However, "CRI_FALSE" never returns, except at severe run-time errors where it is difficult to keep the application running.
Even if the return value is "CRI_TRUE", if you exit the main loop and call the end function, the player will end.

Example of use

```
1  void main(void)
2  {
3      CriBool flag = CRI_TRUE;
4
5      /* Initialize */
6      CriAeroPlayer_Initialize();
7
8      /* Main loop */
9      while( flag )
10     {
11         /* Main */
12         flag = CriAeroPlayer_Main();
13     }
14
15     /* Finalize */
16     CriAeroPlayer_Finalize();
17
18     return;
19 }
```

CriAeroPlayer_Finalize()

Overview

Exit the player

Format

void CRIAPI CriAeroPlayer_Finalize();

Argument

None

Return value

None

Explanation

Quit the player and stop the Aeropoint runtime system

Example of use

```
void main(void)
{
    CriBool flag = CRI_TRUE;

    /* Initialize */
    CriAeroPlayer_Initialize();

    /* Main loop */
    while( flag )
    {
        /* Main */
        flag = CriAeroPlayer_Main();
    }

    /* Finalize */
    CriAeroPlayer_Finalize();

    return;
}
```

Note

Recalling will return without doing anything.

CriAeroPlayer_Callback()

Overview

Callback function (type) that is received when a message occurs.

Format

typedef CriAeroResponsed (*CriAeroPlayer_Callback)(CriAeroMessage Message,
CriSint32 Param, CriSint32 SoundId, CriSint32 MovieId);

Type

Message (first argument of callback function)

```
typedef enum {
    CRI_AERO_MESSAGE_SLIDE_PRE_CHANGE = 0, /* The display slide is changed after this
    CRI_AERO_MESSAGE_SLIDE_POST_CHANGE,        /* Display slide changed

    CRI_AERO_MESSAGE_BUTTON_PRESS,             /* Button pressed
} CriAeroMessage;
```

Response status (return value of callback function)

```
typedef enum {
    CRI_AERO_RESPONSED_NONE         = 0,           // None

    CRI_AERO_RESPONSED_SLIDE_PRE_CHANGE   = (1 <<  0),  // Display slide will change
    CRI_AERO_RESPONSED_SLIDE_POST_CHANGE  = (1 <<  1),  // Display slide changed
    CRI_AERO_RESPONSED_BUTTON_PRESS       = (1 <<  2),  // Button operation
    CRI_AERO_RESPONSED_PLAY_SOUND         = (1 <<  3),  // Sound playback
    CRI_AERO_RESPONSED_PLAY_MOVIE         = (1 <<  4),  // Video playback
} CriAeroResponsed;
```

Argument

Message : Message type

Param : When message is CRI_AERO_MESSAGE_SLIDE_PRE_CHANGE,
→ The slide number to be displayed next.
When Message is CRI_AERO_MESSAGE_SLIDE_POST_CHANGE,
→ Displayed slide number
When message is CRI_AERO_MESSAGE_BUTTON_PRESS,
→ Button number

SoundId : Sound ID

MovieId : Movie ID

Return value

Depending on the contents of the executed process, the enumerator in "CriAeroResponsed" will be returned as an OR condition. If no valid processing was done, it will be returns "CRI_AERO_RESPONSED_NONE".

Explanation

This function is prepared by the user and first registered in the player. The player makes callbacks as needed during the main processing.

Example of use

```
CriAeroResponsed mycallback( CriAeroMessage Message, CriSint32 Param, CriSint32 SoundId, CriSint32 MovieId )
{
    CriAeroResponsed res = CRI_AERO_RESPONSED_NONE;

    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){
        /* SoundId is valid */
        if( SoundId >= 0 ){
            /* Random play */
            CriAeroSound_PlaySound( rand() % 10 );

            /* Cancel default sound control */
            res |= CRI_AERO_RESPONSED_PLAY_SOUND;
        }
    }

    return res;
}
```

CriAeroPlayer_SetCallback()

Overview

Register the callback function with the player

Format

void CRIAPI CriAeroPlayer_SetCallback(const CriAeroPlayer_Callback callback);

Argument

callback : User callback function
Pass NULL to cancel registration

Return value

None

Explanation

Register or unregister the callback function

Example of use

```
void main(void)
{
    CriAeroPlayer_Initialize();
    CriAeroPlayer_SetCallback( mycallback );
       :
       :
```

7.5.4 Page switching API

Read an arbitrary slide from the SD card, display it, and put it in the running state.

List of functions

function	Content
CriAeroPlayer_ReadSlide()	Specify the page and display it

CriAeroPlayer_ReadSlide()

Overview

Specify the slide and display it

Format

CriSint32 CRIAPI CriAeroPlayer_ReadSlide(CriSint32 SlideNo);

Argument

SlideNo : Specify the slide to display with a number from 1 -
Display the start-up screen with -1

Return value

SlideNo : If the display is successful, the same value as the argument is returned.
-1 : Display failure

Explanation

Display the Slide specified by the argument. If the display fails, the startup screen will be displayed.

Fig. 182 Startup screen

]

7.5.5 Video API

Video API plays up to 8 videos set in PowerPoint®.

List of functions

function	Content
CriAeroMovie_Play()	Play the video
CriAeroMovie_Stop()	Stop the video
CriAeroMovie_IsPlay()	Find out if the video is playing

CriAeroMovie_Play()

Overview

Play video.

Format

CriBool CRIAPI CriAeroMovie_Play(CriUint32 MovieId);

Argument

MovieId : Specify the video to play with a number from 0 to 7

Return value

CRI_TRUE : Playback success

CRI_FALSE : Playback failed

Explanation

Starts playing the video specified by the argument

CriAeroMovie_Stop()

Overview

Stop the video

Format

void CRIAPI CriAeroMovie_Stop();

Argument

None

Return value

None

Explanation

If there is a video being played, it will stop

CriAeroMovie_IsPlay()

Overview

Find out if the video is playing.

Format

CriBool CRIAPI CriAeroMovie_IsPlay();

Argument

None

Return value

CRI_TRUE	:	Playing
CRI_FALSE	:	Stopping

Explanation

Find out if the video is playing

7.5.6 Sound API

Sound API plays up to 32 sounds set in PowerPoint®. There are two types of sound, SE and BGM, which can be played simultaneously.

List of functions

function	Content
CriAeroSound_PlaySound()	Play the sound
CriAeroSound_StopSE()	Stop SE
CriAeroSound_StopBGM()	Stop BGM
CriAeroSound_SetVolumeSE()	Sets the SE volume
CriAeroSound_SetVolumeBGM()	Set the BGM volume

CriAeroSound_PlaySound()

Overview

Play the sound

Format

CriBool CRIAPI CriAeroSound_PlaySound(CriUint32 SoundId);

Argument

SoundId : Specify the sound to play with a number from 0 to 31

Return value

CRI_TRUE : Playback success

CRI_FALSE : Playback failed

Explanation

Starts playing the sound specified by the argument. Sounds are automatically assigned to SE or BGM. If it is already playing, it will be interrupted and the newly specified sound will be played.

CriAeroSound_StopSE() / CriAeroSound_StopBGM()

Overview

Stops SE/BGM playback

Format

```
void CRIAPI CriAeroSound_StopSE();
void CRIAPI CriAeroSound_StopBGM();
```

Argument

None

Return value

None

Explanation

SE or BGM stops during playback

CriAeroSound_SetVolumeSE() / CriAeroSound_SetVolumeBGM()

Overview

Specify the SE/BGM volume

Format

```
void CRIAPI CriAeroSound_SetVolumeSE( CriUint32 Volume );
void CRIAPI CriAeroSound_SetVolumeBGM( CriUint32 Volume );
```

Argument

Volume : Specify the volume of SE or BGM (0-100)

Return value

None

Explanation

Specify the volume of SE or BGM

7.5.7 The character display API

The character display API displays characters on the screen by specifying up to 16 fonts set in PowerPoint®.

List of functions

Function	Content
CriAeroFont_Print()	Displays characters on the screen with the specified font
CriAeroFont_Clear()	Clear the display of characters

CriAeroFont_Print()

Overview

Displays characters on the screen with the specified font

Format

CriBool CRIAPI CriAeroFont_Print(CriSint32 X, CriSint32 Y,
CriSint32 FontId, CriAerolColorID ColorId, const char *pText);

Argument

X、Y	:	The XY coordinates to display
FontId	:	Specify the font ID (0 to 15)
ColorId	:	Specify the color
pText	:	The string to display. Specify in UTF-8 code (up to 3 byte code). The maximum length of the character string is 64 bytes.

Return value

CRI_TRUE	:	Display success
CRI_FALSE	:	Display failure

Explanation

Display the string anywhere on the screen.
The character string is displayed in front of other display objects.
The Aeropoint runtime keeps the content specified by this function and redraws it if necessary.

Example of use

```
CriAeroFont_Print( 10, 20, 0, CRI_AERO_COLOR_WHITE, "Hello, world!" );
```

CriAeroFont_Clear()

Overview

Clear the display of characters

Format

void CRIAPI CriAeroFont_Clear();

Argument

None

Return value

None

Explanation

Clears all displayed characters

7.5.8 Display switching API

The display switching API specifies the ID of the shape set in PowerPoint® and switches between display and non-display.

List of functions

Function	Content
CriAeroPlayer_SetShapeVisible ()	Show or hide for the shape with the specified ID

CriAeroPlayer_SetShapeVisible ()

Overview

Show or hide for the shape with the specified ID

Format

```
CriSint32 CRIAPI CriAeroPlayer_SetShapeVisible
        ( CriSint32 ShapeId, CriBool IsVisible );
```

Argument

ShapeId	:	Specifies the ID of the shape to set
IsVisible	:	Specifies whether to show or hide the shape. Displayed with CRI_TRUE, hidden with CRI_FALSE.

Return value

CRI_TRUE	:	Display success
CRI_FALSE	:	Display failure

Explanation

Display the string anywhere on the screen. The character string is displayed before other display objects.

The Aeropoint runtime keeps the content specified by this function and redraws it if necessary.

7.5.9 Test IO API

The "Test IO API" sets the output and gets the input to the test GPIO.

List of functions

Function	Content
CriAeroTestIO_SetLed ()	Sets the output of the specified LED
CriAeroTestIO_CheckLed()	Check the output status of the specified LED
CriAeroTestIO_IsSwitch()	Gets the switch status

CriAeroTestIO_SetLed ()

Overview

Sets the output of the specified LED

Format

void CriAeroTestIO_SetLed(CriAeroTestIOLedColor led, CriBool bit);

Argument

led : Specifies the LED to set

bit : The value to set for the LED
CRI TRUE lights up and CRI FALSE goes out

Return value

None

Explanation

Set to the LED specified by the argument and control lighting and extinguishing.

CriAeroTestIO_CheckLed ()

Overview

Check the output status of the specified LED.

Format

CriBool CriAeroTestIO_CheckLed(CriAeroTestIOLedColor led);

Argument

Led : Specify the LED to get the output status

Return value

CRI_TRUE : The specified LED is lit

CRI_FALSE : The specified LED is off

Explanation

Acquires the output status of the LED specified by the argument and returns the status information of lighting or extinguishing.

CriAeroTestIO_IsSwitch ()

Overview

Gets the switch status

Format

CriBool CriAeroTestIO_IsSwitch();

Argument

None

Return value

CRI_TRUE	:	The switch is not pressed
CRI_FALSE	:	The switch is pressed

Explanation

Gets the state of the switch and returns information about whether it is pressed.

7.5.10 Debug Text API

By using the debug text API, you can display characters on the screen without using font data.

List of functions

Function	Content
CriAeroDebugText_SetFreeMode()	Switch to free mode
CriAeroDebugText_SetConsoleMode()	Switch to console mode
CriAeroDebugText_FreePrint()	Displays characters on the screen in free mode.
CriAeroDebugText_ConsolePrint()	Display characters on the screen in console mode
CriAeroDebugText_Clear()	Clear the display of characters

CriAeroDebugText_SetFreeMode()

Overview

Switch to free mode

Format

void CRIAPI CriAeroDebugText_SetFreeMode(void);

Argument

None

Return value

None

Explanation

Set the debug text output mode to free mode. In free mode, you can display characters by specifying coordinates with CriAeroDebugText_FreePrint().

The default at the start of the program is free mode, so use it when returning from console mode to free mode.

When switching to free mode, the characters already displayed are erased.

Example of use

```
1    CriAeroDebugText_SetFreeMode();
```

CriAeroDebugText_SetConsoleMode()

Overview

Switch to console mode.

Format

void CRIAPI CriAeroDebugText_SetConsoleMode(CriSint32 X, CriSint32 Y
CriUint16 lineLimit);

Argument

X,Y	:	The lower left XY coordinate of the rectangular range used as the console.
lineLimit	:	Maximum number of lines handled as console

Return value

None

Explanation

Sets the debug text output mode to console mode.

In console mode, you can use CriAeroDebugText_ConsolePrint() to display a character on the console.

When switching to console mode, the characters already displayed will be erased.

Example of use

```
CriAeroDebugText_SetConsoleMode(240, 0, 17);
```

CriAeroDebugText_FreePrint()

Overview

Displays characters on the screen in free mode.

Format

CriBool CRIAPI CriAeroDebugText_FreePrint(CriSint32 X, CriSint32 Y,
CriAerolColorID ColorId, CriAerolColorID BackgroundColorId,
const char *pText);

Argument

X,Y	:	XY coordinates to display
ColorId	:	Specify a color
BackgroundColorId	:	Specify background color
pText	:	The string to display Specify with ASCII printable characters (0x20 to 0x7E). The maximum length of the character string is 64 bytes.

Return value

CRI_TRUE	:	Display success
CRI_FALSE	:	Display failure

Explanation

Display the string anywhere on the screen. The character string is displayed in front of other display objects. The Aeropoint runCRItime retains the content specified by this function and redraws it if necessary.
You can display up to 17 lines, and if you execute it with the maximum number of lines displayed, the display will fail. Display always fails when used in console mode.

Example of use

```
CriAeroDebugText_FreePrint( 10, 20, CRI_AERO_COLOR_WHITE, CRI_AERO_COLOR_BLACK, "Hello, world!" );
```

CriAeroDebugText_ConsolePrint()

Overview

Displays characters on the screen in console mode.

Format

CriBool CRIAPI CriAeroDebugText_ConsolePrint(CriAerolColorID ColorId,
CriAerolColorID BackgroundColorId, const char *pText);

Argument

ColorId	:	Specify a color
BackgroundColorId	:	Specify background colPlayback Successor
pText	:	The string to display Specify with ASCII printable characters (0x20-0x7E) The maximum length of the character string is 64 bytes.

Return value

CRI_TRUE	:	Display success
CRI_FALSE	:	Display failure

Explanation

Displays the character string on the screen like a console. The character string is displayed in front of other display objects.

The Aeropoint runtime keeps the content specified by this function and redraws it if necessary.

When you try to display a character in console mode, it will be displayed at the top of the free area. It can hold up to the number of lines set by CriAeroDebugText_SetConsoleMode(). If you execute it with the maximum number of lines displayed, the oldest line will be deleted automatically.

When used in free mode, the display always fails.

Example of use

```
CriAeroDebugText_ConsolePrint( 10, 20, CRI_AERO_COLOR_WHITE, CRI_AERO_COLOR_BLACK, "Hello, world!" );
```

CriAeroDebugText_Clear()

Overview

Clears all displayed characters.

Format

void CRIAPI CriAeroDebugText_Clear(void);

Argument

None

Return value

None

Explanation

Clears all displayed characters.

Example of use

```
CriAeroDebugText_Clear();
```

7.5.11 Primitive display API

"Primitive display API" displays simple figures and lines prepared as a test on the screen.

List of function.

Function	Content
CriAeroPrimitiveDraw_EndPoint()	Displays a simple figure.
CriAeroPrimitiveDraw_Line()	Display a straight line.
CriAeroPrimitiveDraw_Bar()	Display a rectangle.
CriAeroPrimitiveDraw_Clear()	Clears the primitive display.

CriAeroPrimitiveDraw_EndPoint()

Overview

Displays a simple figure.

Format

void CRIAPI CriAeroPrimitiveDraw_EndPoint(CriAeroEndPointType Type,
CriSint32 X, CriSint32 Y, CriAeroColorID Color);

Type

Display figure (first argument)

```
typedef enum {
        CRI_AERO_ETYPE_RECTANGLE = 0,       // Rectangle
        CRI_AERO_ETYPE_CIRCLE,              // Circle
        CRI_AERO_ETYPE_DIA,                 // Diamond
        CRI_AERO_ETYPE_TRIANGLE,            // Triangle
} CriAeroEndPointType;
```

Argument

Type	:	Specify the shape to display.
X、Y	:	The XY coordinates of the figure to display.
Color	:	Specify the color.

Return value

None

Explanation

The specified figure is displayed at arbitrary coordinates on the screen.

The size of the figure is 6x6.

The Aeropoint runtime retains the content specified by this function and redraws it if necessary.

CriAeroPrimitiveDraw_Line()

Overview

Display a straight line.

Format

```
void CRIAPI CriAeroPrimitiveDraw_Line(CriSint32 X1, CriSint32 Y1,
        CriSint32 X2, CriSint32 Y2, CriAeroColorID Color );
```

Argument

X1、Y1	:	XY coordinate of the start point of the line to display
X2、Y2	:	XY coordinate of the end point of the line to display
Color	:	Specify a color

Return value

None

Explanation

A straight line connecting the specified two points from the start point to the end point is displayed. The Aeropoint runtime retains the content specified by this function and redraws it if necessary.

CriAeroPrimitiveDraw_Bar()

Overview

Display a rectangle

Format

```
void CRIAPI CriAeroPrimitiveDraw_Bar( CriSint32 X1, CriSint32 Y1,
        CriSint32 X2, CriSint32 Y2, CriAeroColorID Color );
```

Argument

X1、Y1	:	XY coordinates of the rectangle vertices to display
X2、Y2	:	XY coordinates on the diagonal of X1 and Y1
Color	:	Specify a color

Return value

None

Explanation

Displays a rectangle with two specified points as the diagonal.

Aeropoint runtime keeps the content specified in this function and redraws it if necessary.

CriAeroPrimitiveDraw_Clear()

Overview

Clear primitive display

Format

void CRIAPI CriAeroPrimitiveDraw_Clear();

Argument

None

Return value

None

Explanation

Clears all visible primitives

7.5.12 Data Flash API

Using the data flash API, you can write to the data flash
You can use the area obtained by the function to get data flash information. Data is written in blocks.
The data written to the flash is saved even when the power is turned off, so it can be used for application save data, etc.

List of functions

Function	Description
CriAeroDataFlash_QueryAboutDataFlash ()	Get data flash information
CriAeroDataFlash_Write ()	Write data to flash

CriAeroDataFlash_QueryAboutDataFlash ()

Overview

Get data flash information

Format

```
void CriAeroDataFlash_QueryAboutDataFlash(
CriUint32 *StartAddress,CriUint32 *BytesBlock,CriUint32 *NumBlocks);
```

Argument

StartAddress	:	The start address of the flash is returned
BytesBlock	:	The number of bytes in a block is returned
NumBlocks	:	The total number of blocks in the flash is returned

Return value

None

Explanation

Get the information of the user data area on the flash, which can be written in the unit obtained by BytesBlock.

CriAeroDataFlash_Write ()

Overview

Write data to flash

Format

CriUint32 CriAeroDataFlash_Write(CriUint32 SrcAddress , CriUint32 DestAddress,CriUint32 NumBlocks);

Argument

SrcAddress	:	Specify the address from which to write
DestAddress	:	Specifies the address to write to, in blocks from StartAddress.
NumBlocks	:	Indicates the amount of writes in blocks

Return value

Returns 0 on success. Otherwise, it is an error.

Explanation

Write data to flash

7.5.13 Serial communicaiton API

Serial communication API enables serial communication with another device.
The following functions are used to send UART commands written in "7.5.2 UART Command" and to receive logs.

List of function.

Function	Content
CriAeroSerial_Start()	Start serial communication
CriAeroSerial_Stop()	Stop serial communication
CriAeroSerial_ChangeVolumeBgm()	Send a command to change the background music volume to a specified value
CriAeroSerial_ChangeVolumeSe()	Send a command to change the SE volume to a specified value
CriAeroSerial_SendMoviePlay()	Send a command to play the specified video
CriAeroSerial_SendMovieStop()	Send a command to stop the video
CriAeroSerial_SendScrUpdate()	Send a command to redraw the screen
CriAeroSerial_SendShapeOff()	Send a command to hide the specified shape
CriAeroSerial_SendShapeOn()	Send a command to display the specified shape
CriAeroSerial_SendShowTitle()	Send a command to display the title
CriAeroSerial_SendSlideNo()	Send a command to display the specified slide
CriAeroSerial_SendSoundPlay()	Sends a command to play the specified sound
CriAeroSerial_SendSoundStop()	Send a command to stop the sound
CriAeroSerial_SendTestDemo()	Send the command
CriAeroSerial_SendTextClear()	Send a command to erase the text
CriAeroSerial_SendTextPrint()	Send a command to display the specified text
CriAeroSerial_LogString()	Send a log message
CriAeroSerial_SetLogCallback()	Register a callback function when a log is received

CriAeroSerial_Start()

Overview

Start serial communication

Format

void CriAeroSerial_Start(void);

Argument

None

Return value

None.

Explanation

Start serial communication.

Call this function before the function that sends the command.

CriAeroSerial_Stop()

Overview

Stop serial communication

Format

void CriAeroSerial_Stop(void);

Argument

None

Return value

None.

Explanation

Stop serial communication.

CriAeroSerial_ChangeVolumeBgm() / CriAeroSerial_ChangeVolumeSe()

Overview

Send a command to change the background music volume to a specified value via serial communication

Format

```
void CriAeroSerial_ChangeVolumeBgm(CriUint32 Vol);
void CriAeroSerial_ChangeVolumeSe(CriUint32 Vol);
```

Argument

Vol	:	Specifies the volume from 0 to 255

Return value

None.

Explanation

Send the command to change the volume of BGM or SE to the specified value via serial communication.

Execute this command after calling CriAeroSerial Start.

CriAeroSerial_SendMoviePlay()

Overview

Send a command to play the specified video via serial communication.

Format

void CriAeroSerial_SendMoviePlay(CriUint32 Id);

Argument

Id : Specifies the number of the sound to be played, from 0 to 31.

Return value

None.

Explanation

Send a command to play the specified video via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendMovieStop()

Overview

Send a command to stop the video via serial communication.

Format

void CriAeroSerial_SendMovieStop(void);

Argument

None

Return value

None.

Explanation

Send a command to stop the video via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendScrUpdate()

Overview

Send a command to redraw the screen via serial communication.

Format

void CriAeroSerial_SendScrUpdate(void);

Argument

None

Return value

None.

Explanation

Send a command to redraw the screen via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendShapeOff()

Overview

Send a command to hide the specified shape via serial communication.

Format

void CriAeroSerial_SendShapeOff(CriUint32 Id);

Argument

Id : Specify the ID of the shape to be hidden

Return value

None.

Explanation

Sends a command to hide the specified shape via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendShapeOn()

Overview

Send a command to display the specified shape via serial communication.

Format

void CriAeroSerial_SendShapeOn(CriUint32 Id);

Argument

Id : Specify the ID of the shape to display

Return value

None.

Explanation

Send a command to display the specified shape via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendShowTitle()

Overview

Send a command to display the title via serial communication.

Format

void CriAeroSerial_SendShowTitle(void);

Argument

None.

Return value

None.

Explanation

Send the command to display the title via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendSlideNo()

Overview

Send a command to display the specified slide via serial communication.

Format

void CriAeroSerial_SendSlideNo(CriUint32 No);

Argument

No : Specify the number of the slide to be displayed

Return value

None.

Explanation

Send a command to display the specified slide via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendSoundPlay()

Overview

Send a command to play the specified sound via serial communication.

Format

void CriAeroSerial_SendSoundPlay(CriUint32 Id);

Argument

Id : Specify the number of the sound to play (0~31)

Return value

None.

Explanation

Send a command to play the specified sound via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendSoundStop()

Overview

Send a command to stop the sound via serial communication.

Format

void CriAeroSerial_SendSoundStop(void);

Argument

None.

Return value

None.

Explanation

Send a command to stop the sound via serial communication.

Execute this after calling CriAeroSerial_Start.

CriAeroSerial_SendTestDemo()

Overview

Send the demo number command via serial communication.

Format

void CriAeroSerial_SendTestDemo(CriUint32 No);

Argument

No : Specify the demo to run by number

Return value

None.

Explanation

Send the command via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendTextClear()

Overview

Send a command to erase the text via serial communication.

Format

void CriAeroSerial_SendTextClear(void);

Argument

None.

Return value

None.

Explanation

Send a command to erase the text via serial communication.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SendTextPrint()

Overview

Send commands for text data via serial communication.

Format

void CriAeroSerial_SendTextPrint(CriSint32 Index, CriSint32 X, CriSint32 Y, CriSint32 FontId, CriSint32 ColorId, const CriChar8 *pText);

Argument

Index	:	Specify the text ID (0-7)
X、Y	:	The XY coordinates to display.
FontId	:	Specify the font ID (0~15)
ColorId	:	Specify the color
pText	:	Character string to be displayed. Specify in UTF-8 code (3-byte code).

Return value

None.

Explanation

Send the command via serial communication.

Execute after calling CriAeroSerial_Start.

CriAeroSerial_LogString()

Overview

Send log messages via serial communication.

Format

void CriAeroSerial_LogString(const CriChar8 *pText);

Argument

pText : The string to send.

Return value

None.

Explanation

Send a log message via serial communication.

The message "log" will be sent before the specified string to be sent.

Execute this command after calling CriAeroSerial_Start.

CriAeroSerial_SetLogCallback()

Overview

Register a callback function to be called when a log is received.

Format

void CriAeroSerial_SetLogCallback(CRI_AERO_SERIAL_LOG_CALLBACK *pFunction);

Type

```
typedef void CRI_AERO_SERIAL_LOG_CALLBACK(CriUint32 Length,const CriChar8 *pText);
```

Argument

pFunction : Callback function.

Return value

None.

Explanation

Register a callback function to be called when a log is received.

7.6 Sample program

Introducing API usage examples with reference to other samples

7.6.1 Switch slides

- Project
  - Sample\Project\AeropointRX72N_Sample_SlideRead.zip
- GUI data
  - Sample\GUI\Sample_SlideRead.pptx

You import the above project into "e² studio". You write the GUI data to the SD card by referring to "5.1.1 "Write All"".

It displays three slides every 3 seconds, in order from 1 to 2 to 3 to 1.... The main loop takes 1/60th of a second to go around, so 3 seconds will elapse in 180 loops.

```
/* Main loop */
while( flag )
{
    /* Main */
    flag = CriAeroPlayer_Main();

    flame_count ++;
    /* 3second */
    if( flame_count >= 180 ){

        /* Read next slide */
        slide_index = (slide_index % 3) + 1;
        CriAeroPlayer_ReadPage(slide_index);

        flame_count = 0;
    }

}
```

Update page number and switch slides

Fig. 183 Sample code for slide switching

When executed, the following screen will be displayed.

Fig. 184 Switching pages

7.6.2 Display characters on the screen

- Project
  - Sample\Project\AeropointRX72N_Sample_TextPrint.zip
- GUI data
  - Sample\GUI\Sample_TextPrint.pptx

Each tap of a button displays text in ascending or descending order.

In the Aeropoint Toolkit, the top button ID is set to 0 and the bottom button ID is set to 1.

```
CriAeroResponsed mycallback( CriAeroMessage Message, CriSint32 Param, CriSint32 SoundId, CriSint32 MovieId )
{
    CriAeroResponsed res = CRI_AERO_RESPONSED_NONE;

    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){
        /* Up Button */
        if( Param == 0 ){
            /* Index count up */
            s_text_index = (s_text_index + 1)%3;

            /* Clear & Print Text */
            CriAeroFont_Clear();
            CriAeroFont_Print( 65, 120, 0, CRI_AERO_COLOR_WHITE, s_print_text[s_text_index]);
        }
        /* Down Button */
        if( Param == 1 ){

            /* Index count down */
            s_text_index = (s_text_index + 2)%3;

            /* Clear & Print Text */
            CriAeroFont_Clear();
            CriAeroFont_Print( 65, 120, 0, CRI_AERO_COLOR_WHITE, s_print_text[s_text_index]);
        }
    }

    return res;
}
```

Update character numbers in normal order and display characters

Update character numbers in reverse order and display characters

Fig. 185 Sample code for text display

When executed, the following screen will be displayed.

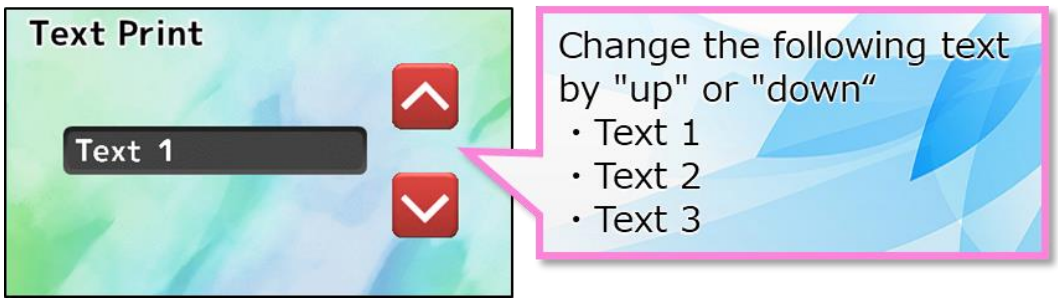


Fig. 186 Application execution screen

7.6.3 Sound playback

- Project
  - Sample\Project\AeropointRX72N_Sample_SoundPlay.zip
- GUI data
  - Sample\GUI\Sample_SoundPlay.pptx

When you run this sample, it adds a play button to the text display to make it sound.
In the Aeropoint toolkit, the play button ID is set to 2.

```
CriAeroResponsed mycallback( CriAeroMessage Message, CriSint32 Param, CriSint32 SoundId, CriSint32 MovieId )
{
    CriAeroResponsed res = CRI_AERO_RESPONSED_NONE;

    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){

        :
        :
        :

        /* Play Button */
        if( Param == 2 ){

            /* Play sound */
            CriAeroSound_PlaySound( s_sound_index );


            /* Cancel default sound control */
            res |= CRI_AERO_RESPONSED_PLAY_SOUND;
        }
    }

    return res;
}
```

Sound Playback

Cancel sound default processing

Fig. 187 Sample code for sound playback

The following screen will be displayed when you run it. Press the play button and the sound will play.

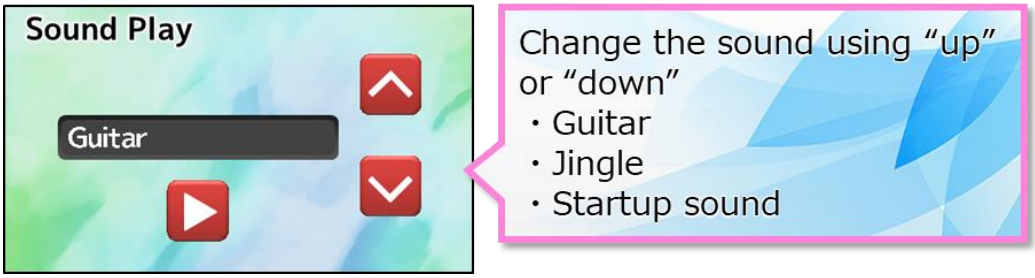


Fig. 188 Application execution screen

7.6.4 Video playback

- Project
  - Sample\Project\AeropointRX72N_Sample_MoviePlay.zip
- GUI data
  - Sample\GUI\Sample_MoviePlay.pptx

When you run this sample, press the play button to play the video.

```
CriAeroResponsed mycallback( CriAeroMessage Message, CriSint32 Param, CriSint32 SoundId, CriSint32 Movie
Id )
{
    CriAeroResponsed res = CRI_AERO_RESPONSED_NONE;

    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){

          :
          :
          :

        /* Play Button */
        if( Param == 2 ){

            /* Play movie */
            CriAeroMovie_Play( s_movie_index );

            /* Cancel default movie control */
            res |= CRI_AERO_RESPONSED_PLAY_MOVIE;
        }
    }

    return res;
}
```

Video playback

Cancel movie default processing

Fig. 189 Video playback

The following screen will be displayed when you run it. The video will play when you select with the up and down buttons and press the play button.

図 190 Application execution screen

7.6.5 Switch visible / invisible

- Project
  - Sample\Project\AeropointRX72N_Sample_ShapeSwitch.zip
- GUI data
  - Sample\GUI\Sample_ShapeSwitch.pptx

The shape is visible and invisible every 3 seconds.

The main loop takes 1/60 seconds per loop, so 180 loop takes 3 seconds.

The shape ID of the circular shape is set to 1.

```
1   /* Main loop */
2   while( flag )
3   {
4       /* Main */
5       flag = CriAeroPlayer_Main();
6
7       flame_count ++;
8       /* 3second */
9       if( flame_count >= 180 ){
10
11          /* Change shape visible */
12          if(shape_visible == CRI_TRUE)shape_visible = CRI_FALSE;
13          else shape_visible = CRI_TRUE;
14          CriAeroPlayer_SetShapeVisible(1, shape_visible);
15
16          flame_count = 0;
17      }
18  }
```

Set shape to visible / invisible

Fig. 191 Sample code to visible / invisible

When executed, the following screen will be displayed.

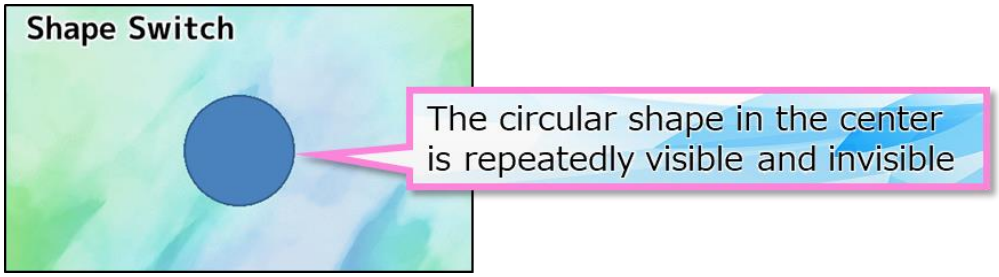


Fig. 192 Application execution screen

7.6.6 Control test IO

- Project
  - Sample\Project\AeropointRX72N_Sample_TestIOSwitch.zip
- GUI data
  - Do not use

The green LED turns on and off every 3 seconds. The main loop takes 1/60 seconds per loop, so 180 loop takes 3 seconds.

```
1   /* Main loop */
2   while( flag )
3   {
4       /* Main */
5       flag = CriAeroPlayer_Main();
6
7       flame_count ++;
8       /* 3second */
9       if( flame_count >= 180 ){
10
11          /* Check led output */
12          if(CRI_TRUE == CriAeroTestIO_CheckLed(CRI_AERO_TESTIO_LED_GREEN)){
13              led_bit = CRI_FALSE;
14          }
15          else {
16              led_bit = CRI_TRUE;
17          }
18
19          /* Setting led output */
20          CriAeroTestIO_SetLed(CRI_AERO_TESTIO_LED_GREEN, led_bit);
21
22          flame_count = 0;
23      }
24
25  }
```

Get LED output information and determines the next output value

Specify LED output information

Fig. 193 Sample code for control test IO

When executed, the following screen will be displayed.

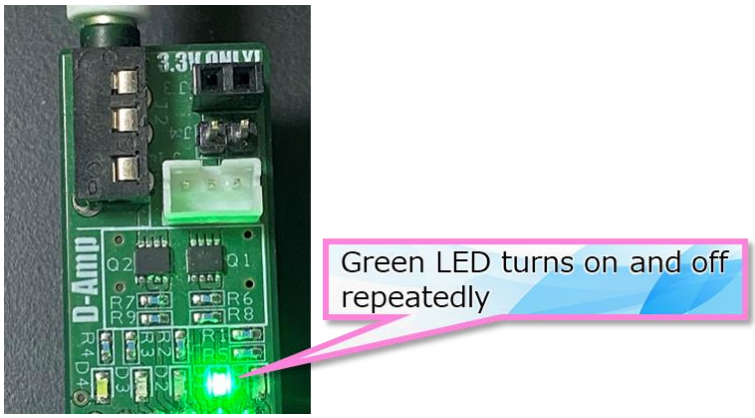


Fig. 194 LED when application execute

7.6.7 Display debug text on screen

- Project
  - Sample\Project\AeropointRX72N_Sample_DebugTextPrint.zip
- GUI data
  - Sample\GUI\Sample_DebugTextPrint.pptx

Each time you tap the right arrow button, the debug text will be displayed in the frame.

Tap the dotted circle button to delete all debug text.

Tapping the Mode button toggles between Free Mode and Console Mode.

In the Aeropoint toolkit, the right arrow button ID is set to 0, the dotted line button ID is set to 1, and the mode button is set to toggle the slide.

```
    /* Button was pressed */
    if( Message == CRI_AERO_MESSAGE_BUTTON_PRESS ){
        /* Print Button */
        if( Param == 0 ){

            /* Print DebugText */
            if(s_mode_index == 0){
                print_res = CriAeroDebugText_FreePrint( 50 + (rand()%150), 90 + (rand()%120), CRI_AERO_COLOR_WHITE, CRI_AERO_COLOR_BLACK, s_print_debugtext[s_debugtext_index]);
            }
            else if(s_mode_index == 1){
                print_res = CriAeroDebugText_ConsolePrint(CRI_AERO_COLOR_WHITE, CRI_AERO_COLOR_BLACK, s_print_debugtext[s_debugtext_index]);
            }

            if(print_res == CRI_TRUE){
                /* Index count up */
                s_debugtext_index = (s_debugtext_index + 1)%17;
            }
        }
        /* Clear Button */
        if( Param == 1 ){
            /* Clear DebugText */
            CriAeroDebugText_Clear();
            s_debugtext_index = 0;
        }
    }

    /* Slide was changed */
    if( Message == CRI_AERO_MESSAGE_SLIDE_PRE_CHANGE ){
        /* Free print Slide */
        if(Param == 1){
            CriAeroDebugText_SetFreeMode();
            s_mode_index = 0;
        }
        /* Console print Slide */
        else if( Param == 2 ){
            CriAeroDebugText_SetConsoleMode(50,90,9);
            s_mode_index = 1;
        }

        s_debugtext_index = 0;
    }
```

Display debug text.
The display function changes depending on the mode.

Remove debug text

Mode switching

Fig. 195 Sample code to display debug text

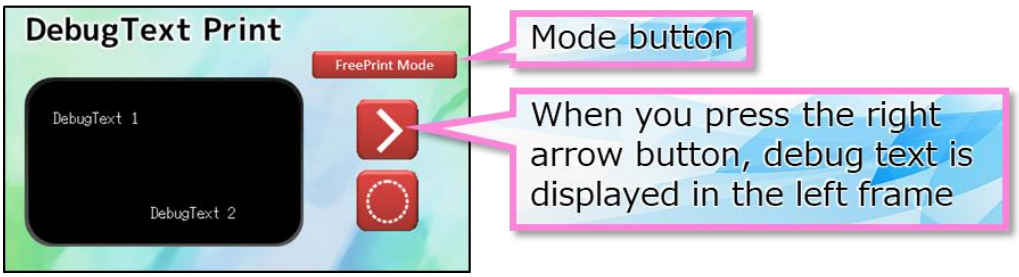


Fig. 196 Application execution screen

7.6.8 Draw a primitive shape on the screen

- Project
  - Sample\Project\AeropointRX72N_Sample_PrimitiveDraw.zip
- GUI data
  - Sample\GUI\Sample_PrimitiveDraw.pptx

It displays a different colored rectangle every second and repeats the process of clearing everything in 4 seconds. The main loop takes 1/60th of a second to complete, so 1 second will pass in 60 loops.

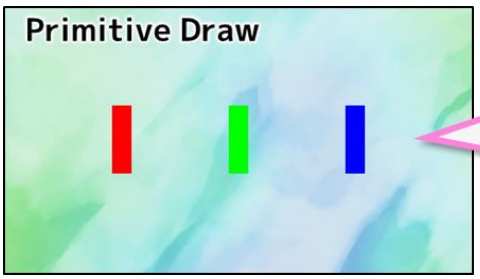
```
1   /* Main loop */
2   while( flag )
3   {
4       /* Main */
5       flag = CriAeroPlayer_Main();
6
7       flame_count ++;
8       /* 1second */
9       if( flame_count >= 60 ){
10
11          /* Primitive draw */
12          if(primitive_count == 1){
13              CriAeroPrimitiveDraw_Bar( 110, 100, 130, 170, CRI_AERO_COLOR_RED);
14          }
15          else if(primitive_count == 2){
16              CriAeroPrimitiveDraw_Bar( 230, 100, 250, 170, CRI_AERO_COLOR_GREEN);
17          }
18          else if(primitive_count == 3){
19              CriAeroPrimitiveDraw_Bar( 350, 100, 370, 170, CRI_AERO_COLOR_BLUE);
20          }
21          else {
22              CriAeroPrimitiveDraw_Clear();
23          }
24
25          primitive_count = (primitive_count+1)%4;
26          flame_count = 0;
27      }
28  }
```

Rectangular Display

Clear the primitive display.

Fig. 197 Sample code for primitive display

When you run it, the following screen will be displayed.



The red, green, and blue rectangles appear in sequence and then repeat the clearing process.

Fig. 198 application execution screen

7.6.9 Serial communication

- Project
  - Sample/Project/AeropointRX72N_Connector.zip
- GUI data
  - Sample/GUI/Sample_Connector.pptx

※For details on how to connect to another device, refer to "5 How to connect serial communication"

No, Id, and command are selected with the buttons and serial communication is performed. The received log is displayed.

```
void main(void)
{
    CriBool flag = CRI_TRUE;

    /* Initialize */
    CriAeroPlayer_Initialize();
    CriAeroCommandQueue_SetSendStatusMode(CRI_FALSE);

    /* Serial Start */
    CriAeroSerial_Start();
    /* Set serial reception log callback */
    CriAeroSerial_SetLogCallback(uartLogCallback);

     :
     :

    /* Main loop */
    while (flag)
    {
        :
        :

        /* Main */
        flag = CriAeroPlayer_Main();
    }

    /* Serial Stop */
    CriAeroSerial_Stop();

    /* Finalize */
    CriAeroPlayer_Finalize();

    return;
}
```

Start serial communication. Register a log receive callback function

Stop serial communication

Fig. 199 Sample code for the start of serial communication

```
 /* Button was pressed */
if (Message == CRI_AERO_MESSAGE_BUTTON_PRESS) {
    switch (Param) {

     :

    /* Send button */
    case 10: {
        switch (nowSlide) {
        case 1:
            /* Page change */
            CriAeroSerial_SendSlideNo(nowNumber[nowSlide]);
            break;
        case 2:
            /* Shape visible */
            CriAeroSerial_SendShapeOn(nowNumber[nowSlide]);
            break;
        case 3:
            /* Sound play */
            CriAeroSerial_SendSoundPlay(nowNumber[nowSlide]);
            break;
        case 4:
            /* Movie play */
            CriAeroSerial_SendMoviePlay(nowNumber[nowSlide]);
            break;
        }
        res = CRI_AERO_RESPONSED_BUTTON_PRESS;
        break;
    }
    case 11: {
        switch (nowSlide) {
        case 2:
            /* Shape unvisible */
            CriAeroSerial_SendShapeOff(nowNumber[nowSlide]);
            break;
        case 3:
            /* Sound stop */
            CriAeroSerial_SendSoundStop();
            break;
        case 4:
            /* Movie stop */
            CriAeroSerial_SendMovieStop();
            break;
        }
        res = CRI_AERO_RESPONSED_BUTTON_PRESS;
        break;
    }
    case 12: {
        switch (nowSlide) {
        case 3:
        case 4:
            /* Volume up */
            volume++;
            if( volume > 4 ){ volume = 4; }
            CriAeroSerial_ChangeVolumeBgm( volval[volume] );
            CriAeroSerial_ChangeVolumeSe(  volval[volume] );
            break;
        }
        res = CRI_AERO_RESPONSED_BUTTON_PRESS;
        break;
    }
    case 13: {

         :

    }
    }
}
```

Send the following commands for each slide
Show the specified slide.
Show the specified shape.
Play the specified sound
Play the specified video

Send the following commands for each slide
Hide the specified shape
Stop sound
Stop video

Send the following commands for each slide
BGM volume setting
SE volume setting

図 200 Sample code for the command transmission part

When you run the program, the following screen will be displayed.

図 201 Application execution screen

8 How to connect to other equipment

By using the UART mechanism, applications can be controlled from the device by connecting to another device and performing serial communication.
When you send out a command from the device on the Master side (such as your MPU), you can control the device on the Slave side (Aeropoint GUI). If necessary, the result of the command will be sent from the device on the slave side to the device on the master side.

Use the terminal CN5 for UART connection. The baud rate is 9600bps.
When connecting two RX72Ns, connect them as shown below.

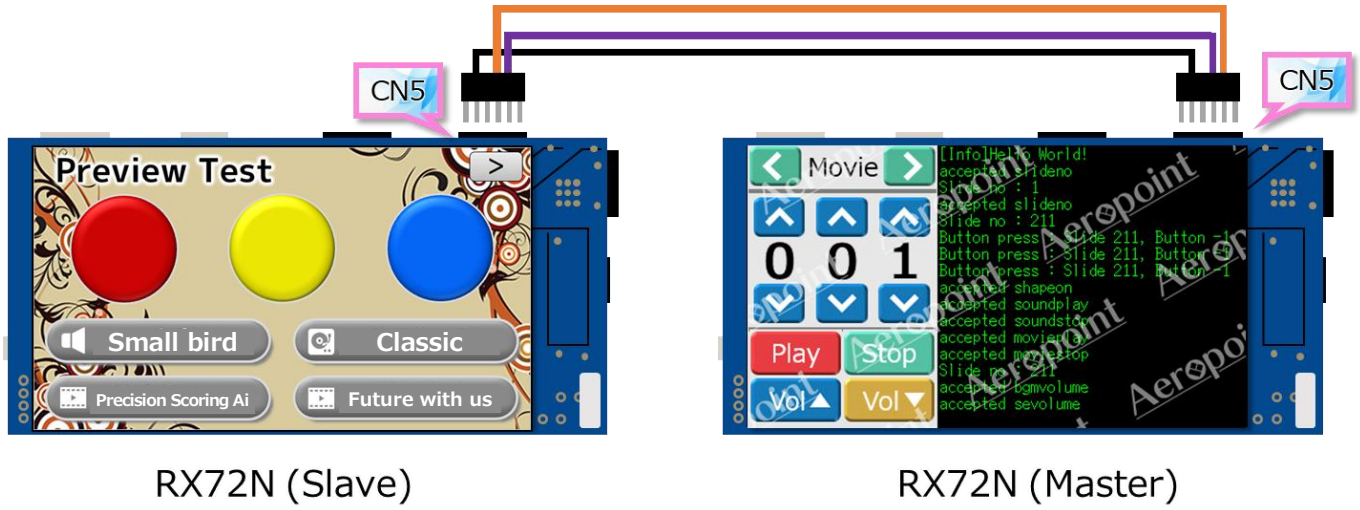


図 202 Connection diagram between RX72N

図 203 Enlarged view of CN5 pin connection

If you want to operate Aeropoint from another device, change the RX72N on the Master side to that device.

9 Make Font

Create font data to display the text on the actual device.

9.1 "FontMaker" tool launch

Select "Make font" from the Options group of the Aeropoint toolkit.

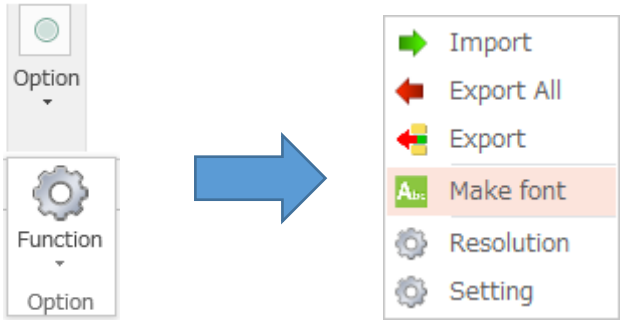


Fig. 204 Option Group menu

"FontMaker" tool will launch.

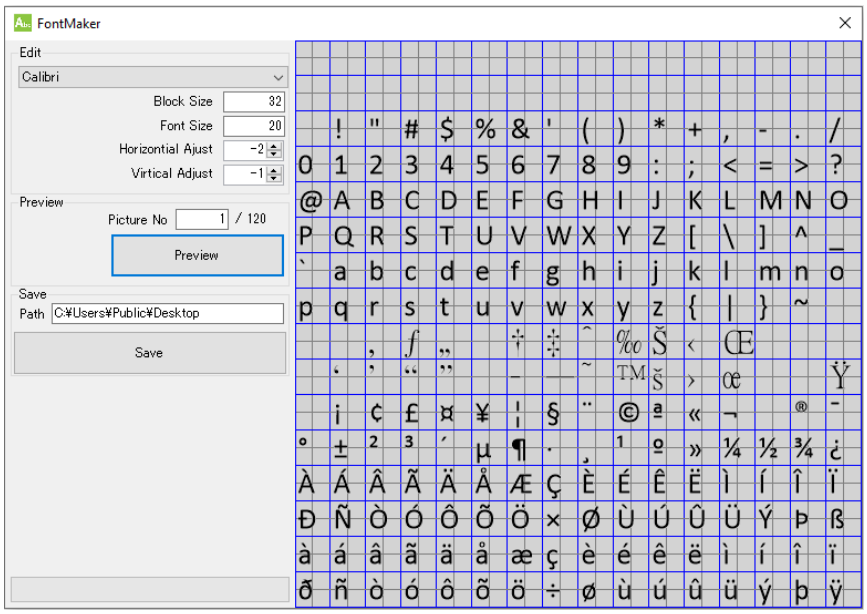


Fig. 205 "FontMaker" tool

9.2 How to use

9.2.1 Pull-down menu

When you open the pull-down menu of the "Edit" group box, a list of fonts that can be used on the PC is registered. If you select one of the fonts, the "Make Font" tool will create data that can be used on the actual device.

> ⚠ Even if it is displayed on the PC, it is not a font that can be used commercially on the actual device. In order to use fonts on the actual device for commercial use, the customer needs to make a contract with the font company independently.

9.2.2 "Size of 1 character block"

In "Size of 1 character block", you specify the size of the block that stores one font in pixels. It is the section marked with a "blue frame" in the right window of the "Make Font" tool.
The specifications of the size of the section are as follows.

- Maximum size of section
  - 60 pixels

9.2.3 "Font size"

Set "Font size" to a value less than "Block size".

9.2.4 Preview

Click on "Preview", the font will be displayed in the right window.
Fine-tune the font, adjusting its size and position, etc. If you change the "Image number", you can check the characters such as double-byte characters.

9.2.5 Position correction

Each character is adjusted by "Horizontal Adjust" and "Vertical Adjust" so that each character is left-aligned within the blue rectangle area. When the characters are displayed side by side on the device, the space between the characters is automatically filled.

9.2.6 Save

When you have finished making adjustments, decide the save destination path and click "Save". It takes a long time to save because of the large number of characters to handle and the amount of calculation. The saved AC1FC format file can be used by writing it on the SD card inserted in the actual device by the method described in "5.2.1"Text"".

10 Program writing

Install "Renesas Flash Programmer" on the PC to write the execution binary of the sample program to the microcomputer.

10.1 Renesas Flash Programmer

Please download and install the following application.

- [Free-of-charge edition] Renesas Flash Programmer V3.06.01
  - https://www.renesas.com/software/D4001218.html

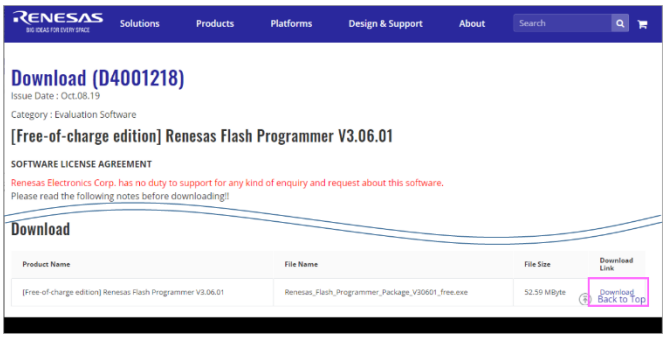


Fig. 206 Download page

10.2 Writing procedure

10.2.1 Change to write mode

Set the DIP switch (SW1 / SW3) of the Envision Kit as shown below.
And connect ECN1 terminal adjacent to SW1 and PC with Micro USB.

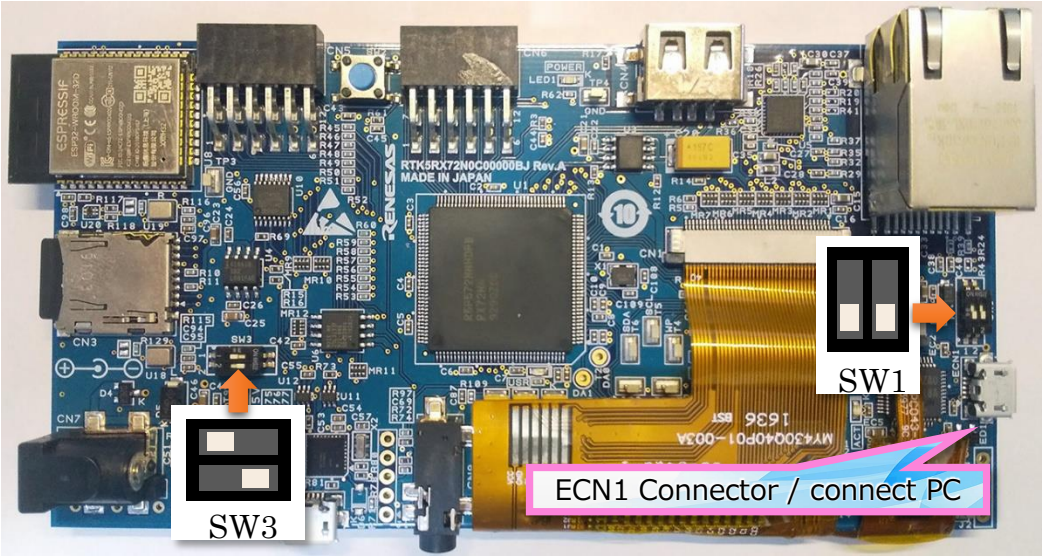


Fig. 207 Set the DIP switch (SW1/SW3) to write setting

10.2.2 Write the program with Renesas Flash Programmer

Launch Renesas Flash Programmer and select New Project from file.

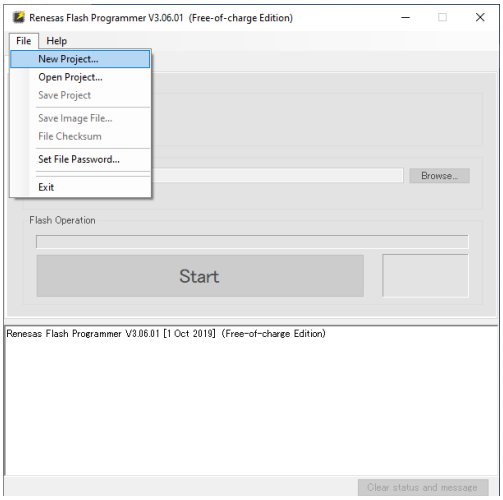


Fig. 208 Select New Project in Renesas Flash Programmer

In Create New Project dialog, set "Microcontroller" to "RX72x", "Tools" to "E2 emulator Lite", "Interface" to "FINE", and click "Connect".

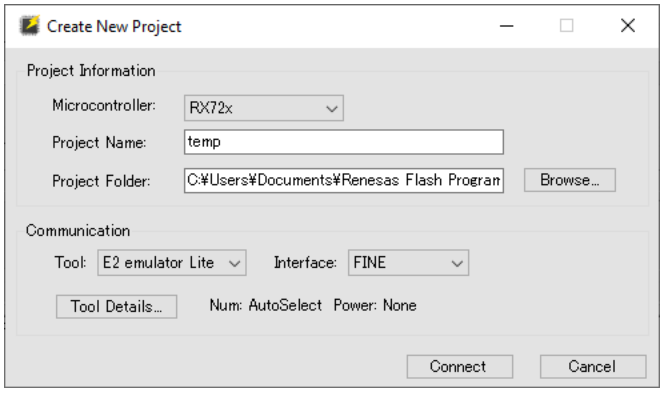


Fig. 209 Create New Project dialog

Enter "F..." in ID Code.

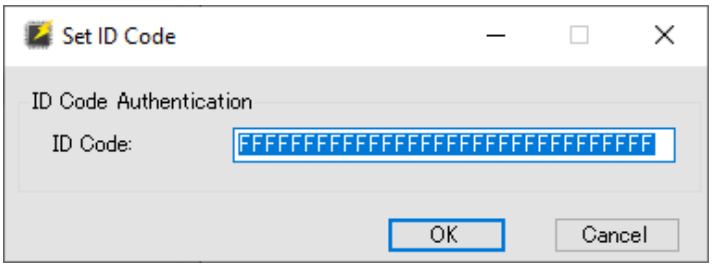


Fig. 210 ID Code setting

To write Aeropoint player, specify "Sample\Player\Player.mot" in "Program File" and click "Start".

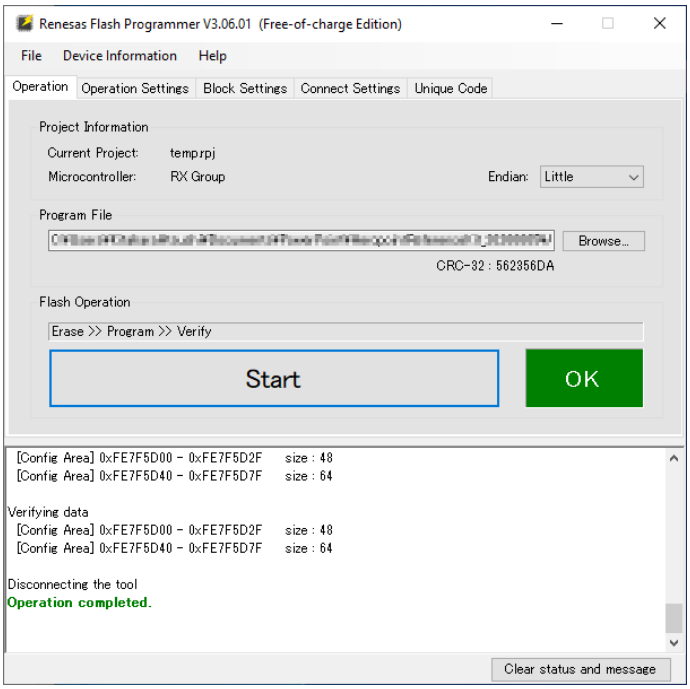


Fig. 211 Program writing

10.2.3 Change to single boot mode

After successfully writing the program, set the DIP switch (SW1) of the Envision Kit as shown below.

Fig. 212 Set the DIP switch (SW1) to single boot

The written program will start.

11 Write data image

The software of Reference kit plays GUI, movies and sounds while reading the data in the microSD card. This chapter describes the procedure to write the attached data image to the microSD card correctly.

11.1 Things to prepare

- microSD Card
  - The data size is about 1GB, so you don't need a large capacity
- microSD Card reader
  - If your PC does not have an SD card slot, prepare a card reader.
- Data image
  - Use "Sample\SDImage\SDImage_xx.img" attached to Reference kit.
- Win32 Disk Imager
  - A tool used to write image data.
    Download from below.
    https://sourceforge.net/projects/win32diskimager/
    ※ Validated using Release 1.0.0.

11.2 Writing procedure

1. Launch Win32 Disk Imager

   Launch Win32 Disk Imager. Administrator rights are required at launch. A message from User Account Control (UAC) is displayed. Click "Yes".

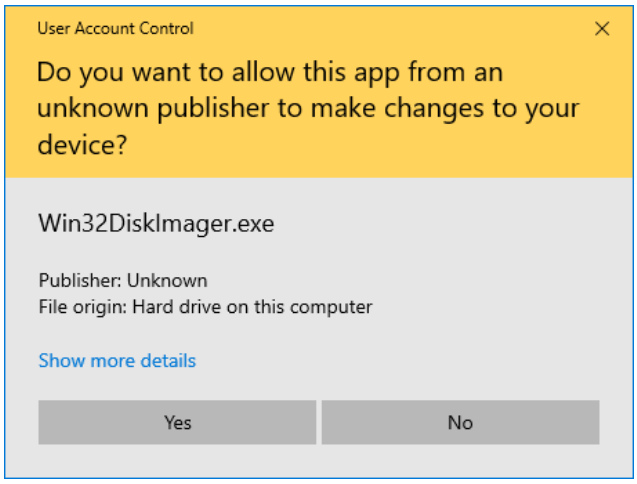


Fig. 213 Message from UAC

2. Confirm insertion of microSD

   Confirm the microSD is properly inserted in your PC (or Card Reader)

3. Check if the drive letter is microSD's

   Check if the drive letter (alphabet) in the device column of the dialog points to the microSD writing device.

   In principle, the drive letter of the built-in DISK is not displayed as a candidate in the Device column, but be careful if multiple removable media are connected.

4. Specify the data image

   You press the icon on the right edge of the Image File column and select the data image (Sample\SDImage\SDImage_xx.bin) to write to the SD card.

5. Write to microSD card

   Confirm all items are set correctly and press the "Write" button.

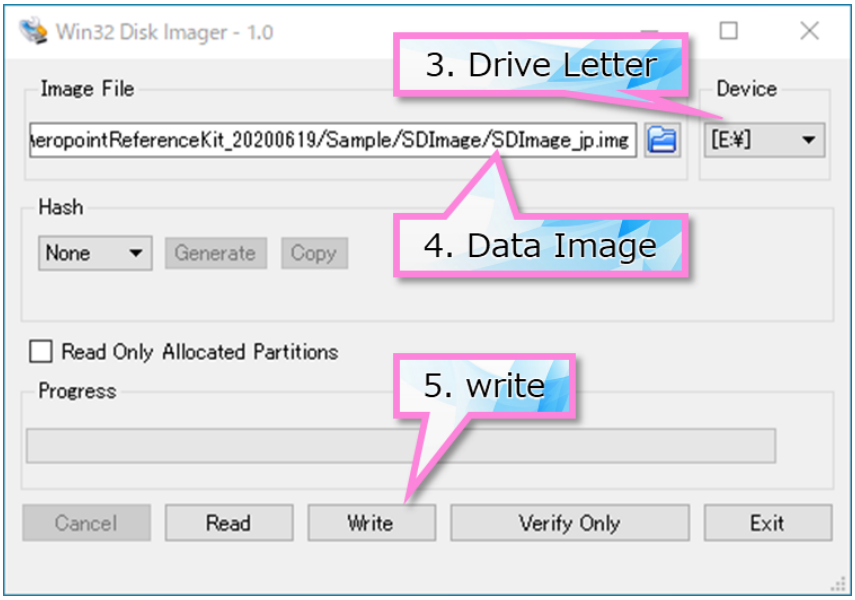


図 214 Win32 Disk Imager screen

12 Q&A

Answers and solutions to common questions and problems.

12.1 Aeropoint toolkit add-in does not appear

If the "AeopointLite" menu does not appear in the menu after installing the Aeropoint toolkit as described in Section "3.2.2 Aeropoint toolkit" , try the following procedure.

1. Select "Options" from the "File" menu bar.

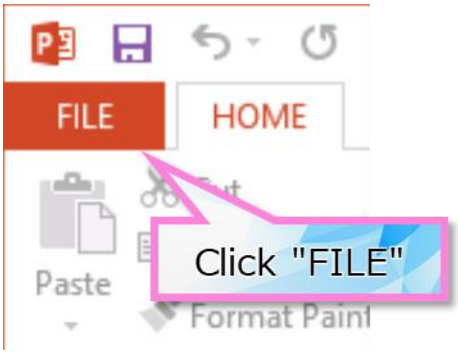


Fig. 215 Click "FILE"

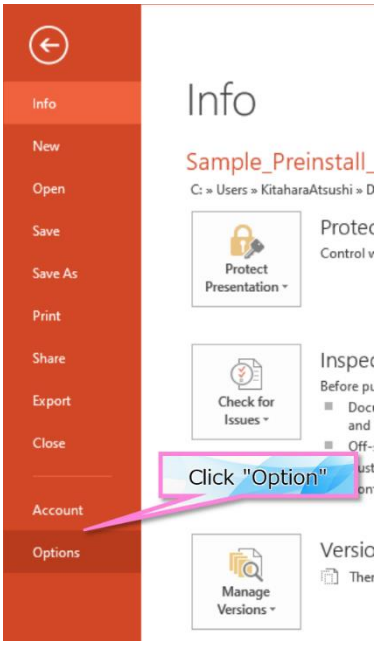


Fig. 216 Select "Option"

2. Select "Add-ins" in the left-hand menu of "PowerPoint® Options" window.

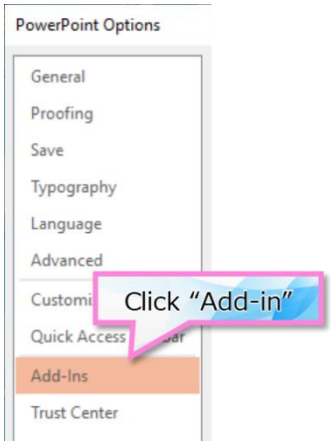


Fig. 217 click "Add-in"

3. Make sure that "AeropointLite32AddIn" is present in "Active Add-Ins".

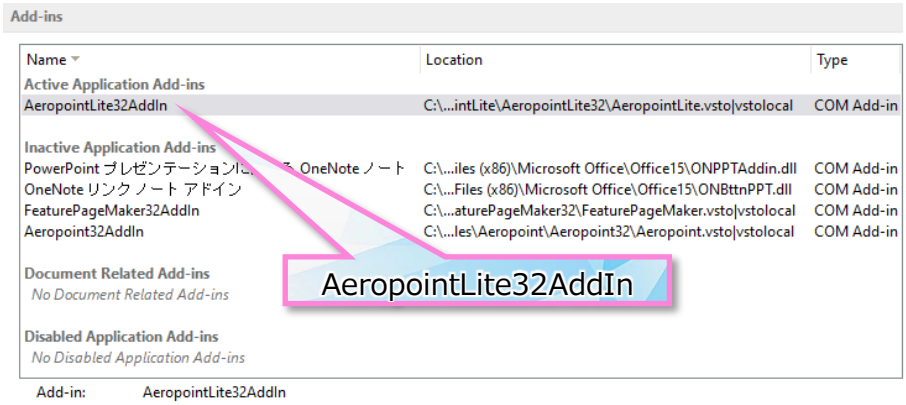


Fig. 218 Example of display when "AeropointLite32AddIn" is present

※ If "AeropointLite32AddIn" is not present in "Active Add-ins", try the following

4. Select "COM Add-in" from "Manage:" page at the bottom of the add-in page, and then click "Go..."

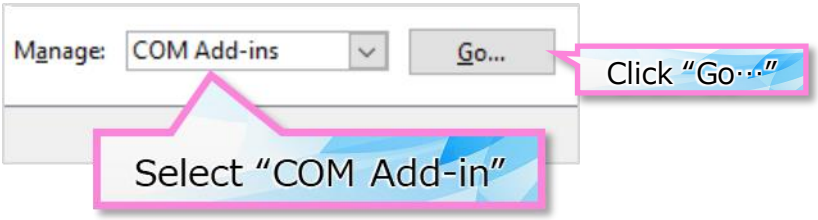


Fig. 219 Select "COM Add-in"

5. Select the "AeropointLite32AddIn" checkbox in the "COM Add-In" dialog and click "OK".

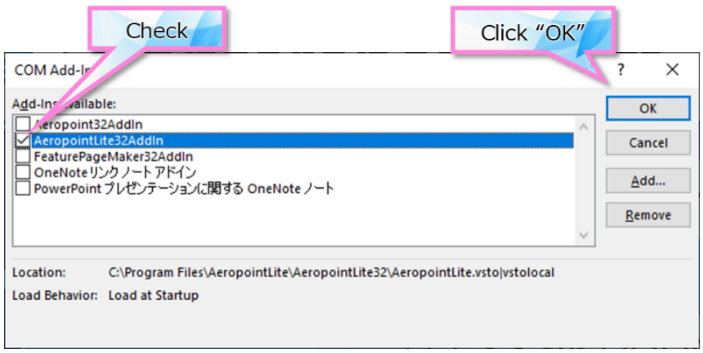


図 220 "COM Add-in" dialog

※At this time, if "AeropointLite32AddIn" is not found in "COM Add-in", please try the following.

6. At the bottom of the add-in page, go to "Manage" and select "Disabled Items" and click "Go...".

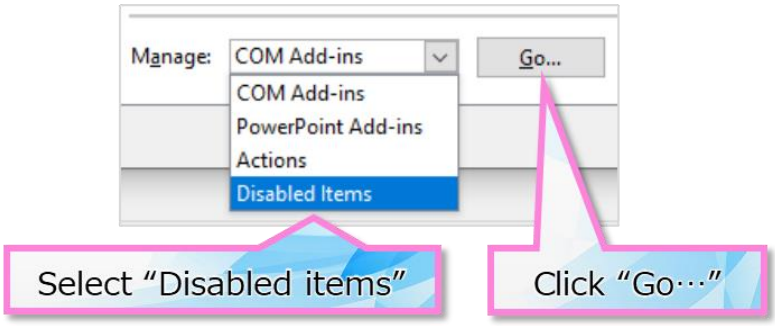


Fig. 221 Select "Disabled items"

7. Select "AeropointLite32AddIn" and click "Enable"

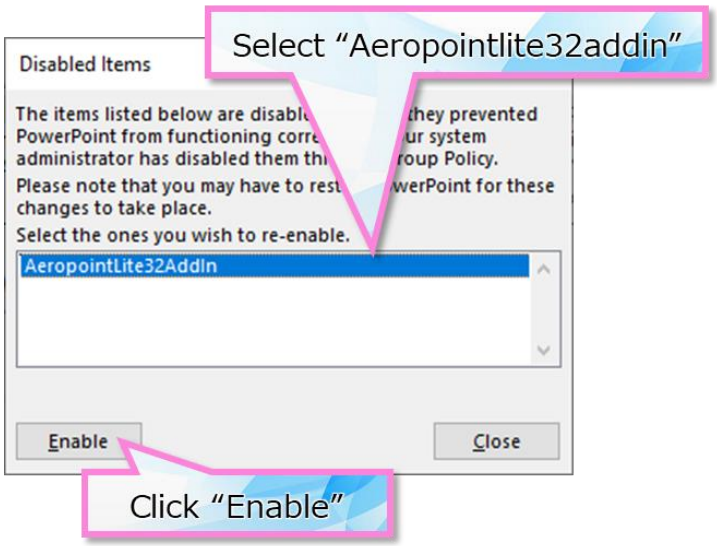


Fig. 222 Select "AeropointLite32AddIn"

8. After enabling "AeropointLite32AddIn", follow the steps 4 to 5. above for setting up "COM add-in"

12.2 When using slide theme or master slide, shapes are not placed correctly on the actual device.

Aeropoint does not support the placement of images in "Slide Themes" or "Master Slides".
This is because it will not be able to correctly calculate the coordinates of shapes placed in a regular slide if images are placed in these mechanisms.

13 Reference price

See below for reference prices.

https://www.cri-mw.co.jp/product/embedded/Aeropoint/GUI_en/price78k1e0000015w53.pdf

14 Contact us

14.1 CRI Middleware

CRI Middleware Co.,Ltd.
〒150-0002
Sumitomo Bldg. 9F, 1-7-7 Shibuya, Shibuya-ku, Tokyo
Email：inquiry-global-emb@cri-mw.co.jp

15 Revision record

Rev.	Date of issue	Revised content
1.00	2020/07/31	First edition issued
1.06	2021/03/15	Added information about Fit kit.