# RX671 Group

## RX671 OTA-supported flat panel HMI PoC with touch keys and LCD

### Introduction

This kit for RX671 touch keys and OTA-supported flat panel HMI PoC using LCD (hereinafter referred to as “RX671 PoC”) is ideal for the development of IoT devices with HMI using capacitive touch sensors.

This application note introduces the HMI solution using RX671 PoC to realize touch functionality and LCD display with serial connection LCD. In addition, introduce the function to connect to the cloud and perform firmware updates using OTA.

The sample program described in this application note is configured using the following libraries.

- LCD Display : Embedded GUI software emWin (hereinafter referred to as “emWin”)

### Target Device

RX671 Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.2

### Target Tool

RX671 PoC

For more information on security, please refer to ”Renesas MCU Firmware Update Design Policy (R01AN5548)”

## Contents

# 1. Outline

This application note describes the operation and structure of the RX671 PoC. RX671 PoC is equipped with touch buttons, touch slider and LCD (240 x 320) and can be used as a demonstration to control the display and settings by imagining a microwave oven. In addition, RX671 PoC can rewrite the program OTA from the cloud.

The overall RX671 PoC image is shown below.

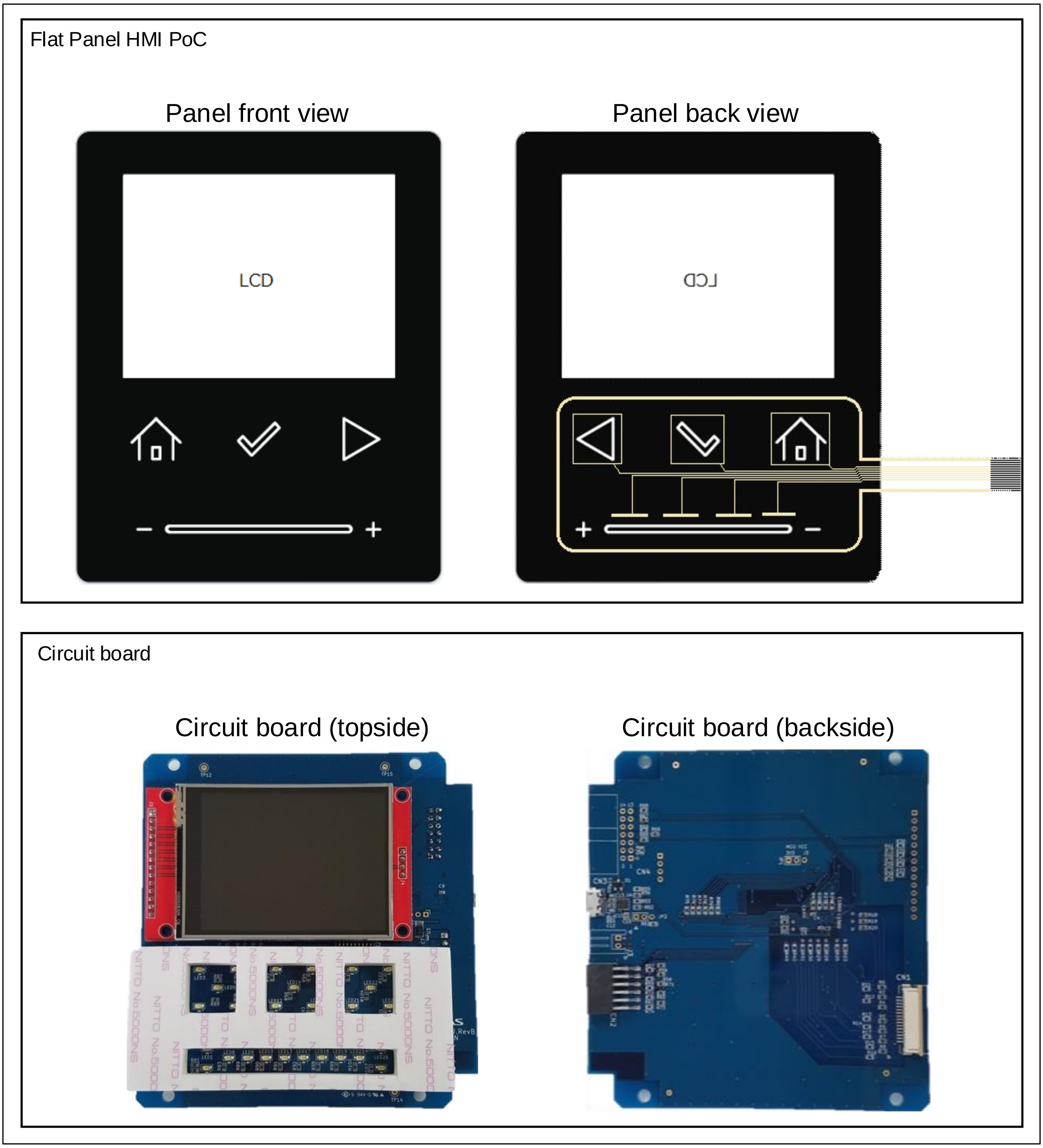


**Figure 1-1 Overall RX671 PoC image**

The system configuration is shown below.

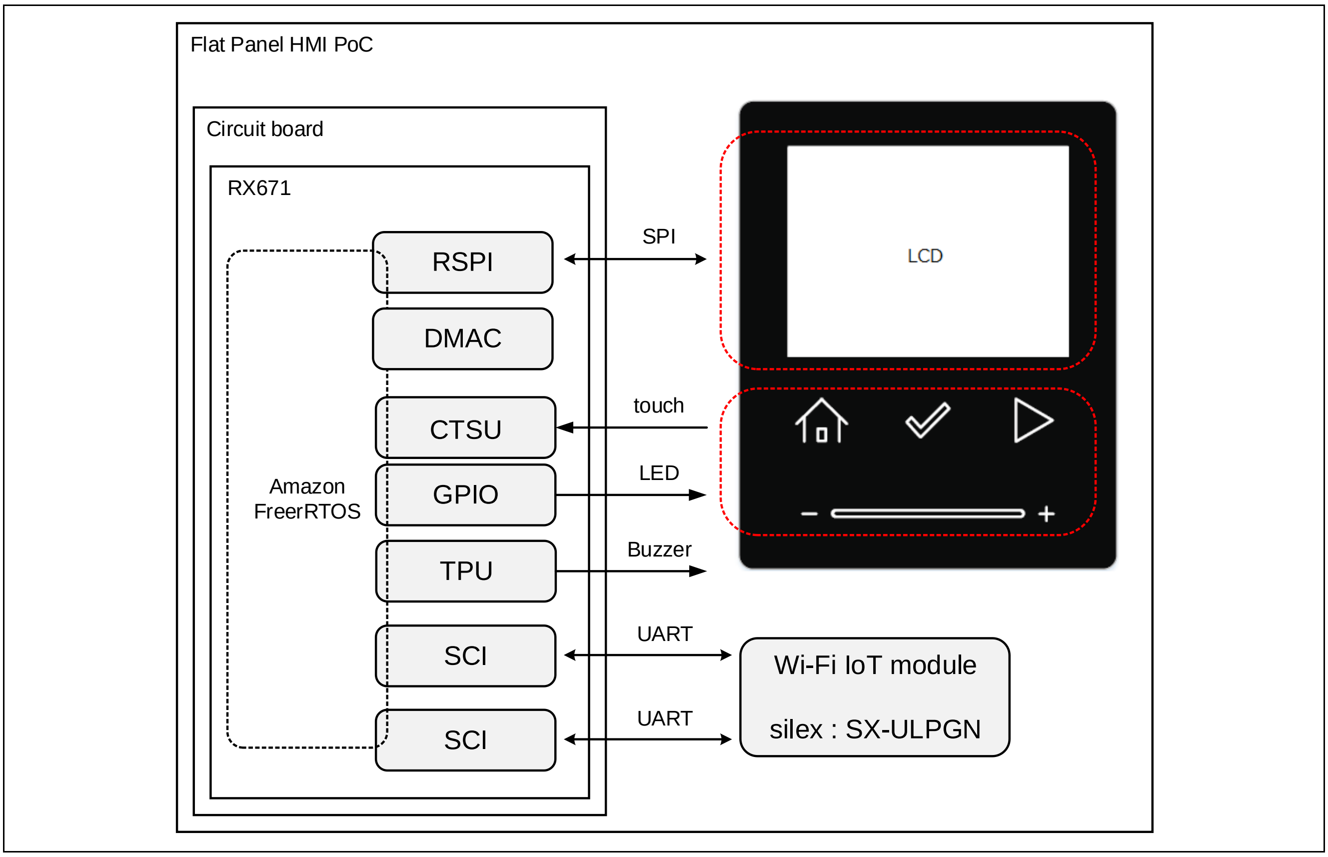


**Figure 1-2 System configuration**

The software configuration is shown below.

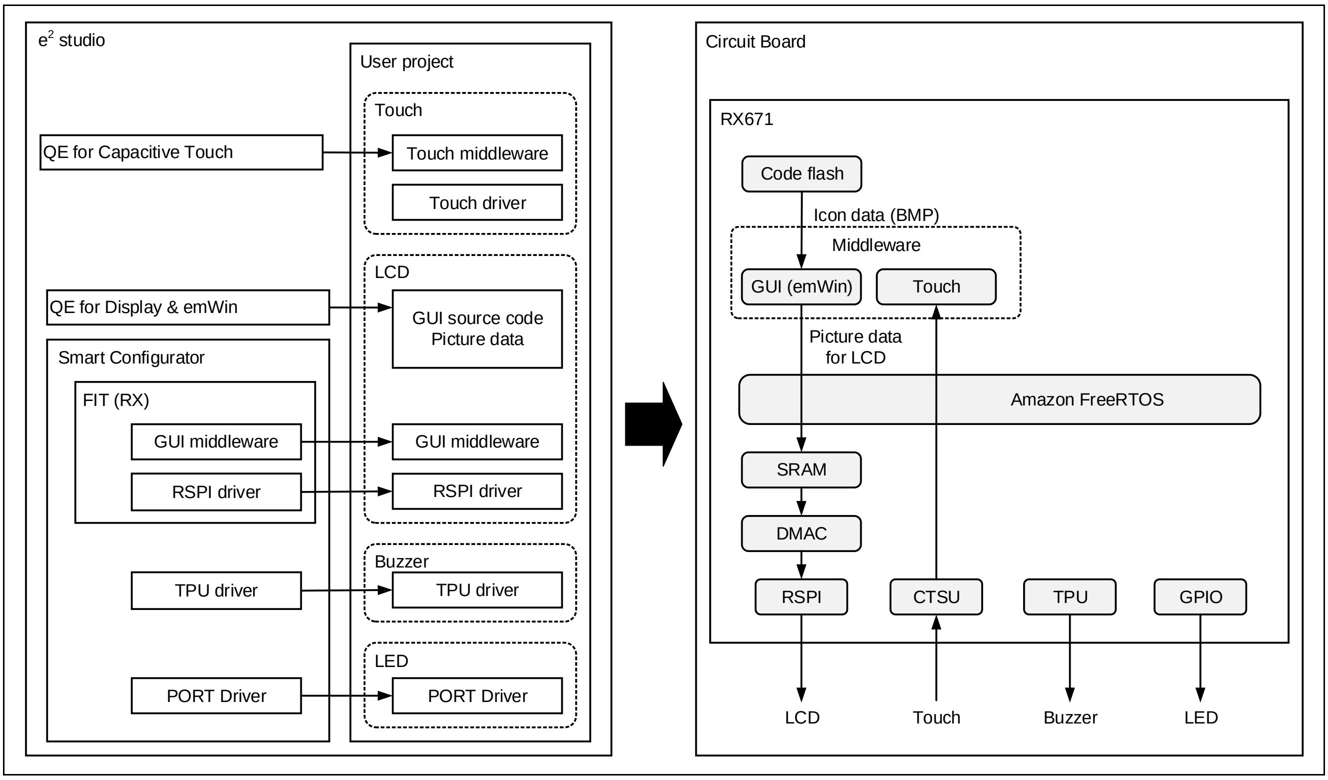


**Figure 1-3 Software configuration**

## 2. Operation Confirmation Conditions

The operation of the sample program has been confirmed under the following conditions.

**Table 2-1 Operation Confirmation Conditions**

| Item | Contents |
|---|---|
| MCU used | R5F5671EHDFM (RX671 Group) |
| Operating frequency | Operating frequency (ICLK) : 120MHz<br>Peripheral operating frequency (PCLKB) : 60MHz |
| Operating voltage | 3.3 V |
| Integrated development | Renesas Electronics<br>e$^2$ studio Version 2023-01 (23.1.0) |
| C compiler | Renesas Electronics<br>C/C++ Compiler Package for RX Family V3.05.00 |
| | Compiler option<br>Default settings of integrated development environment |
| Smart Configurator | V2.16.0 |
| Board support package (r_bsp) | V7.20 |
| Endian order | Little Endian |
| Operating mode | Single chip mode |
| Processor mode | Super visor mode |
| Sample code version | V1.00 |
| Emulator | E2 Emulator Lite |

**Table 2-2 Operation Confirmation Conditions (LCD, Wi-Fi)**

| Item | Contents |
|---|---|
| LCD module | 2.8 TFT SPI 240 × 320 serial port module |
| Wi-Fi module | Wi-Fi-Pmod-Expansion-Board (RTK00WFMX0B00000BE) |

# 3. Sample Programs

## 3.1 Demonstration Screen Flowchart

The demonstration screen flowchart of this sample program is shown below. For detail on each screen, refer to chapter 5.

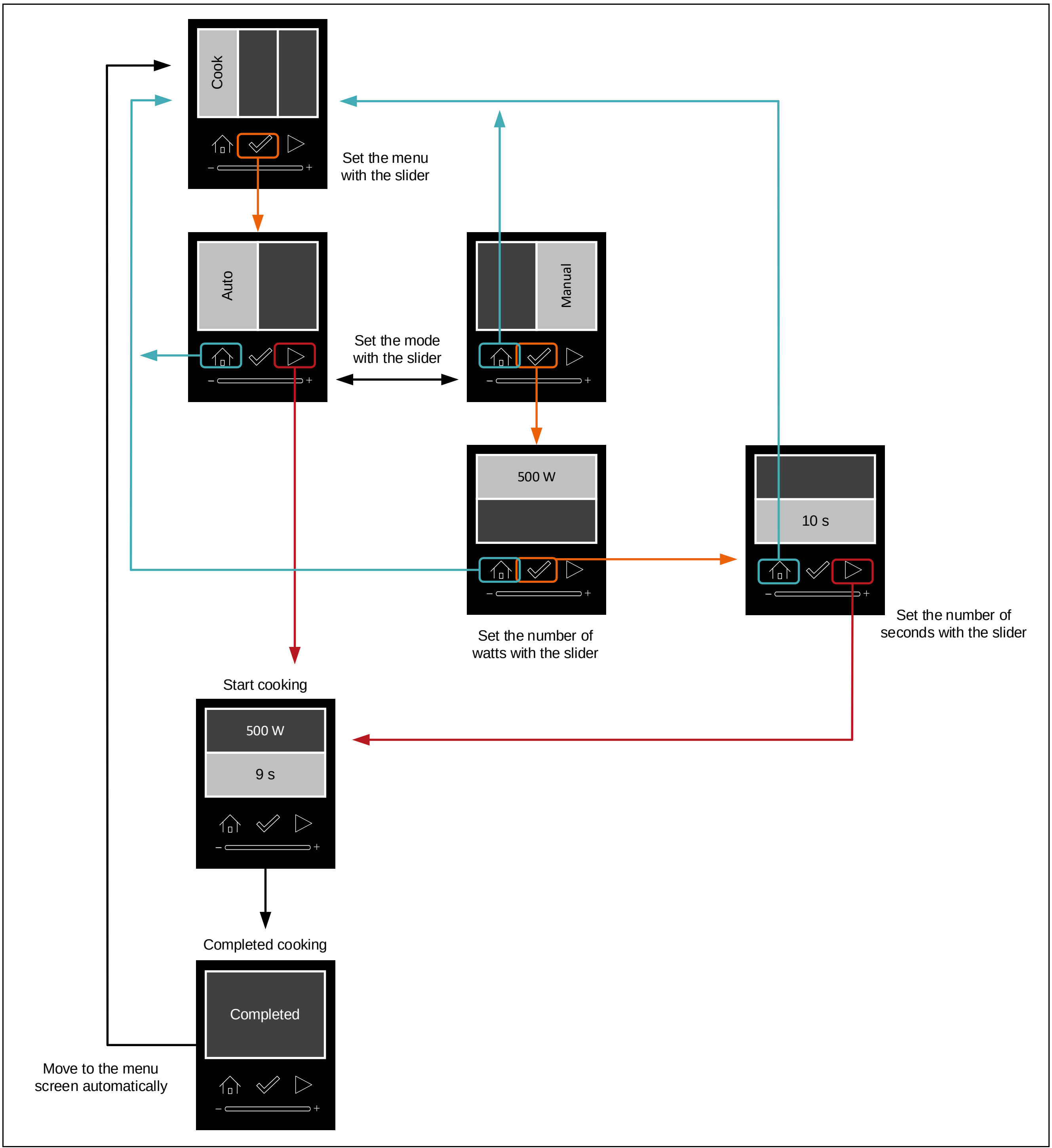


**Figure 3-1 Flowchart of demonstration screen (Cook)**

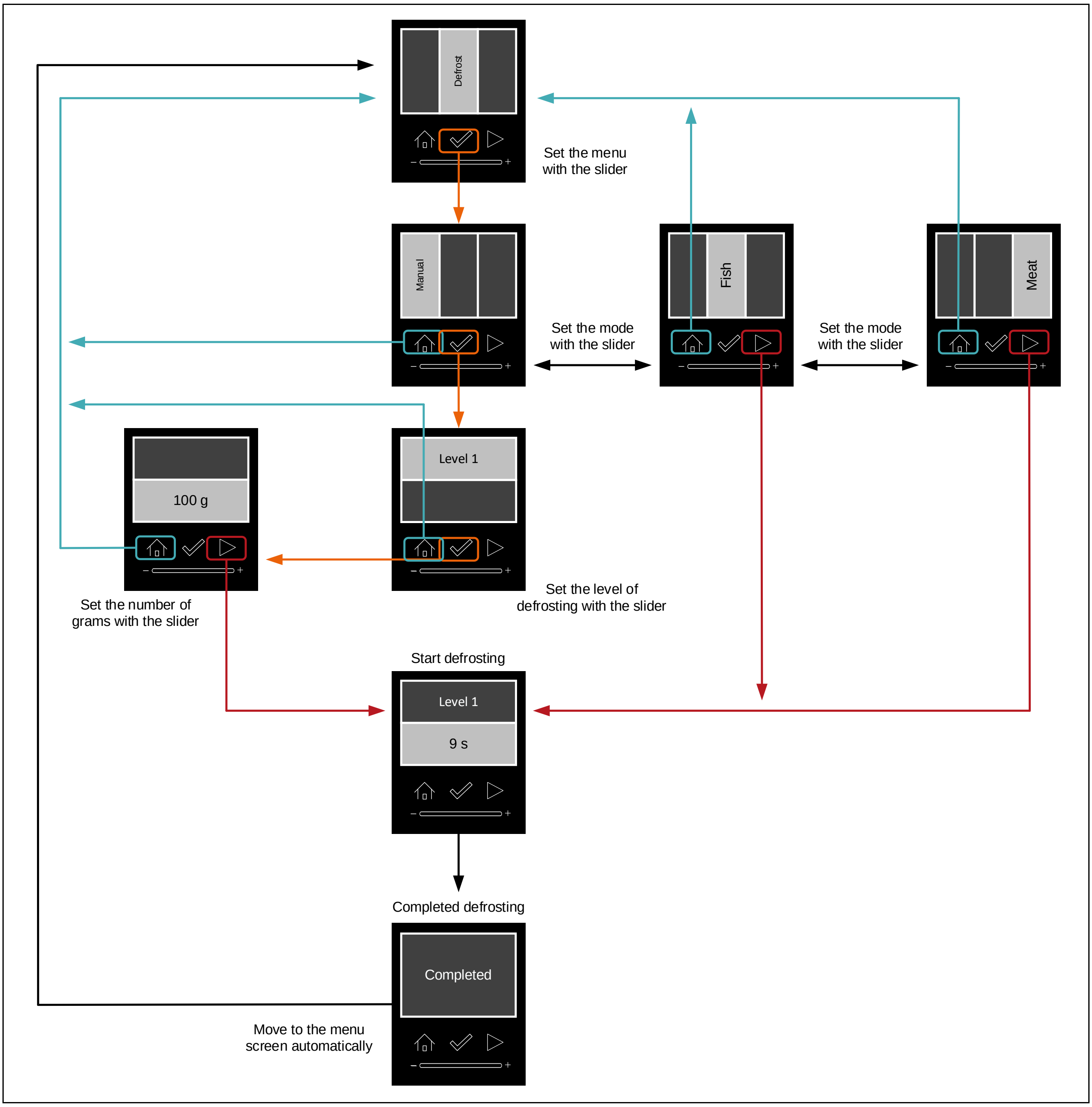


**Figure 3-2 Flowchart of demonstration screen (Defrost)**

Recipe mode is not available with the firmware ver.0.90.

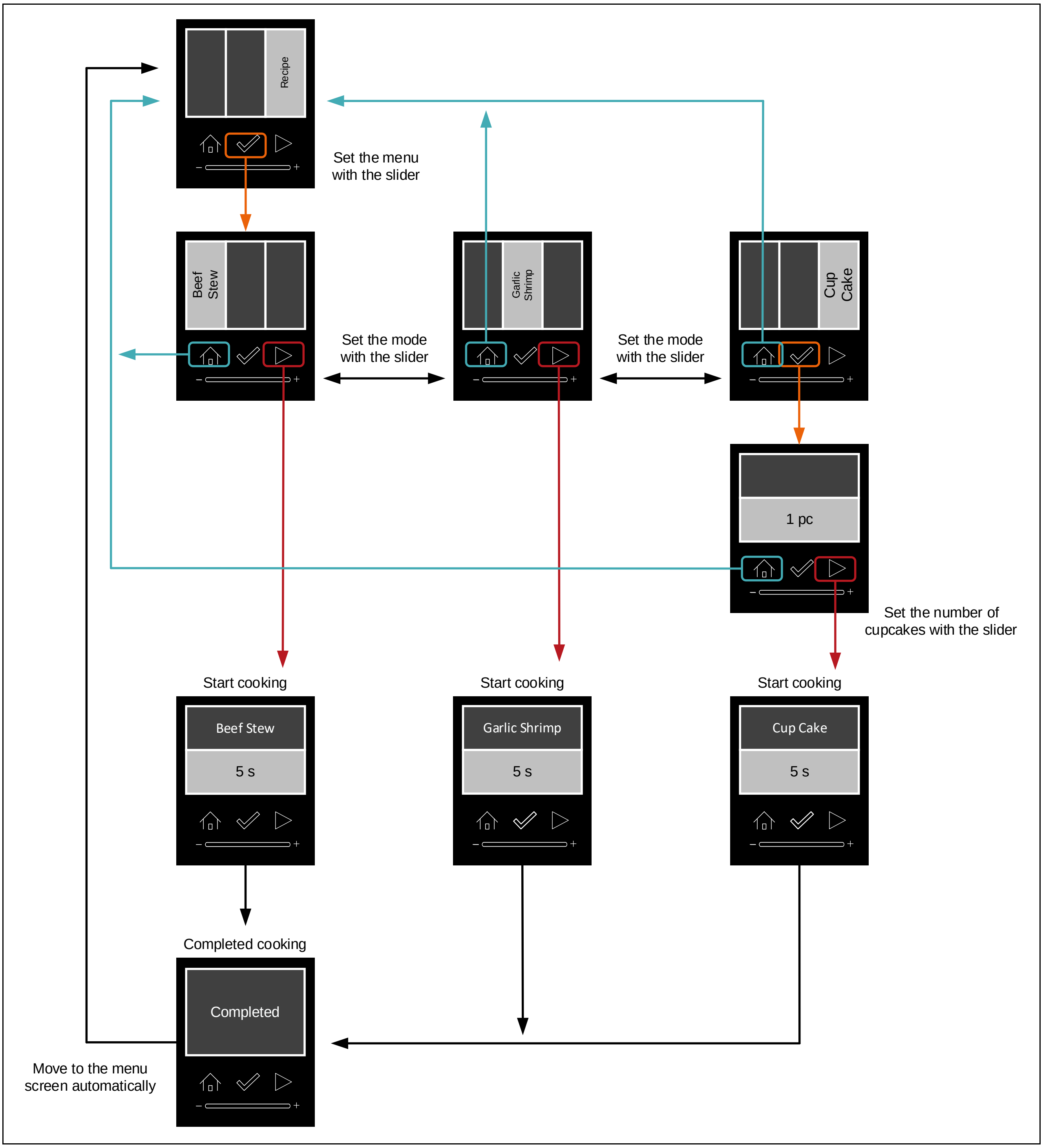


**Figure 3-3 Flowchart of demonstration screen (Recipe)**

## 3.2 Flowchart

### 3.2.1 Overall flowchart of LCD control

The overall flowchart of LCD control is shown below.

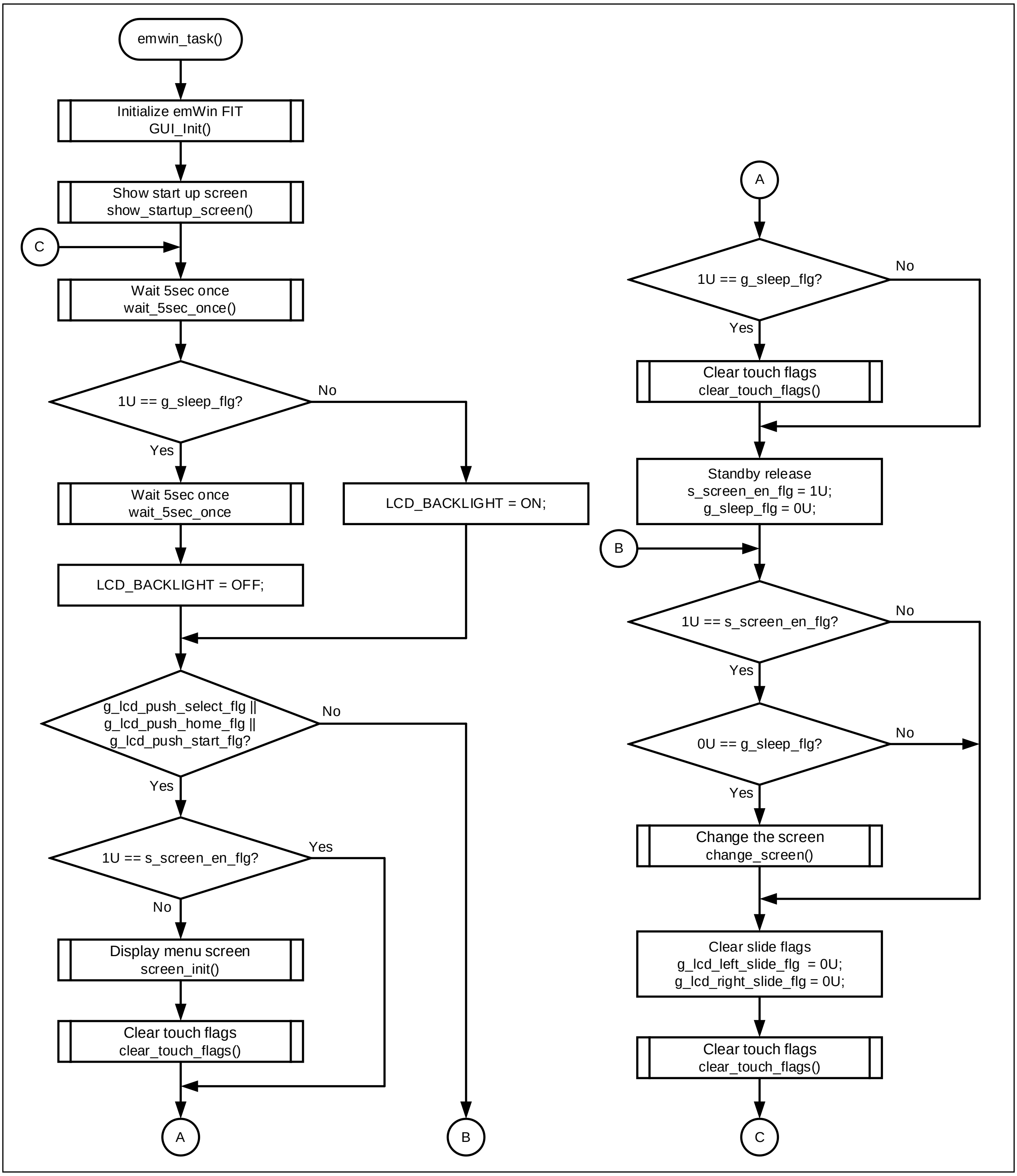


**Figure 3-4 Overall flowchart of LCD control**

### 3.2.2 Processing at touch keys operation

The flowchart for touch keys operation is shown below.

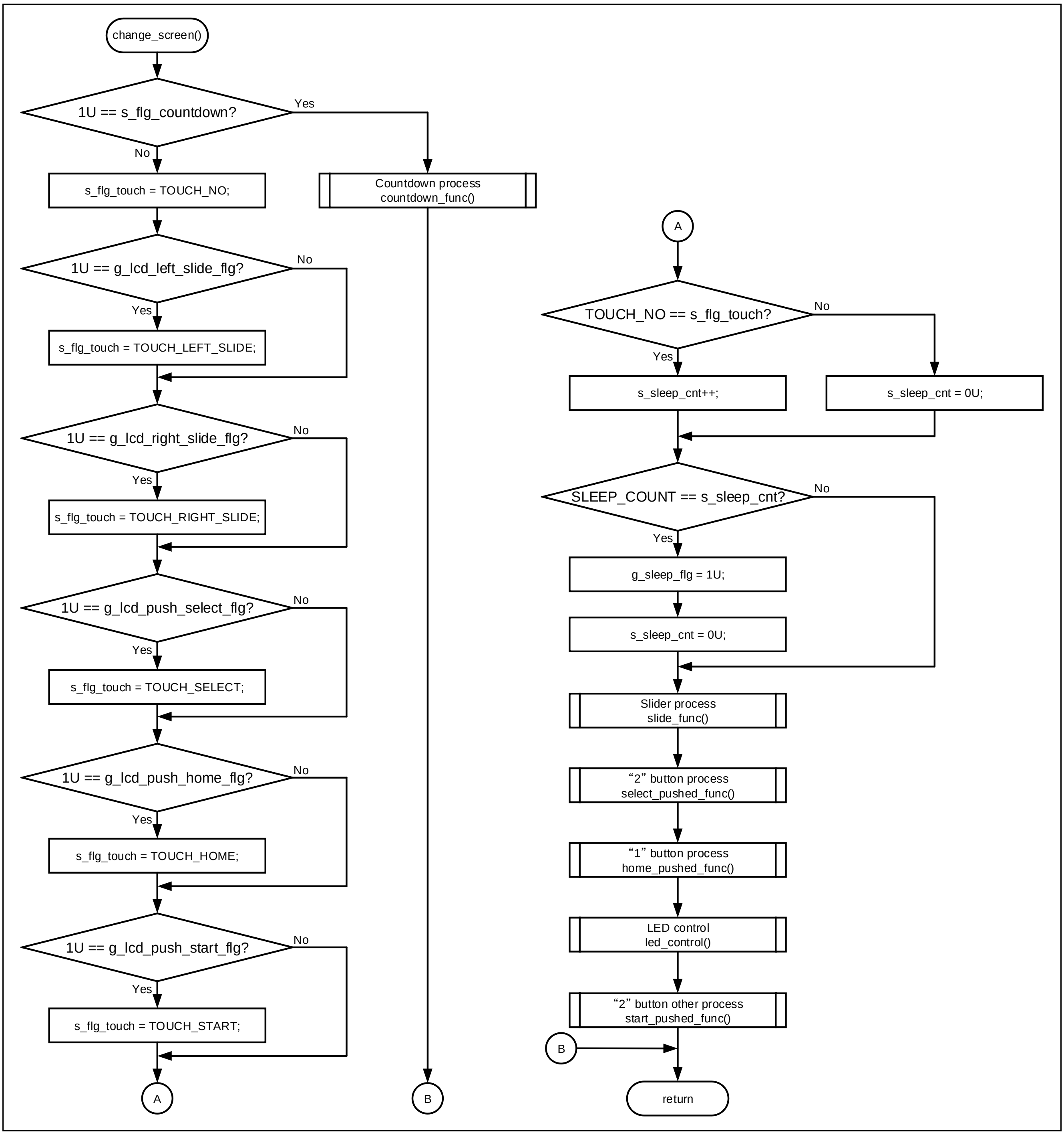


**Figure 3-5 Flowchart for touch keys operation**

### 3.2.3 Processing at touch slider operation

The flowchart for touch slider operation is shown below.

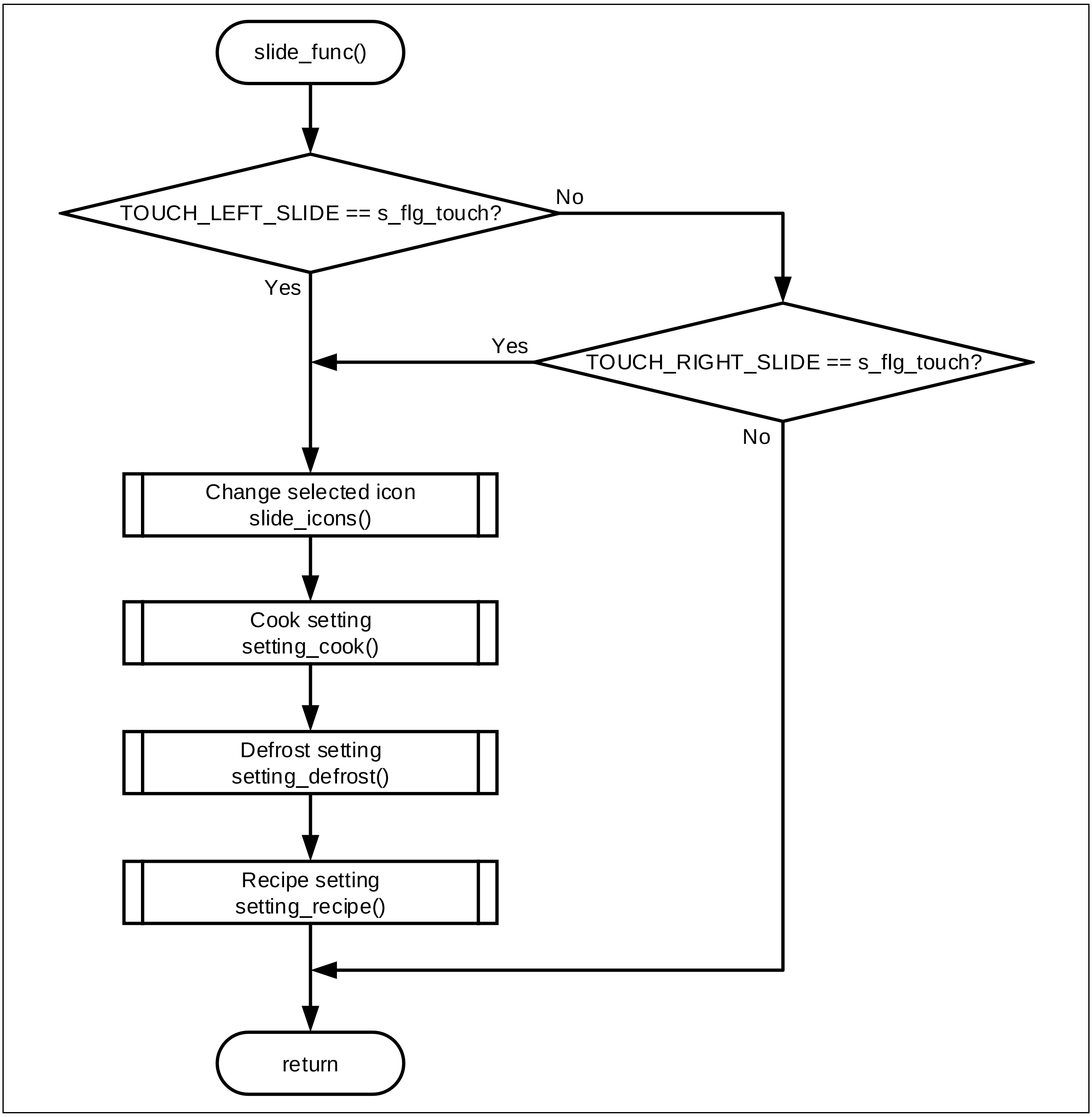


**Figure 3-6 Flowchart for touch slider operation**

### 3.2.4 Processing when the "home" button is touched

The flowchart when the "home" button is touched is shown below.

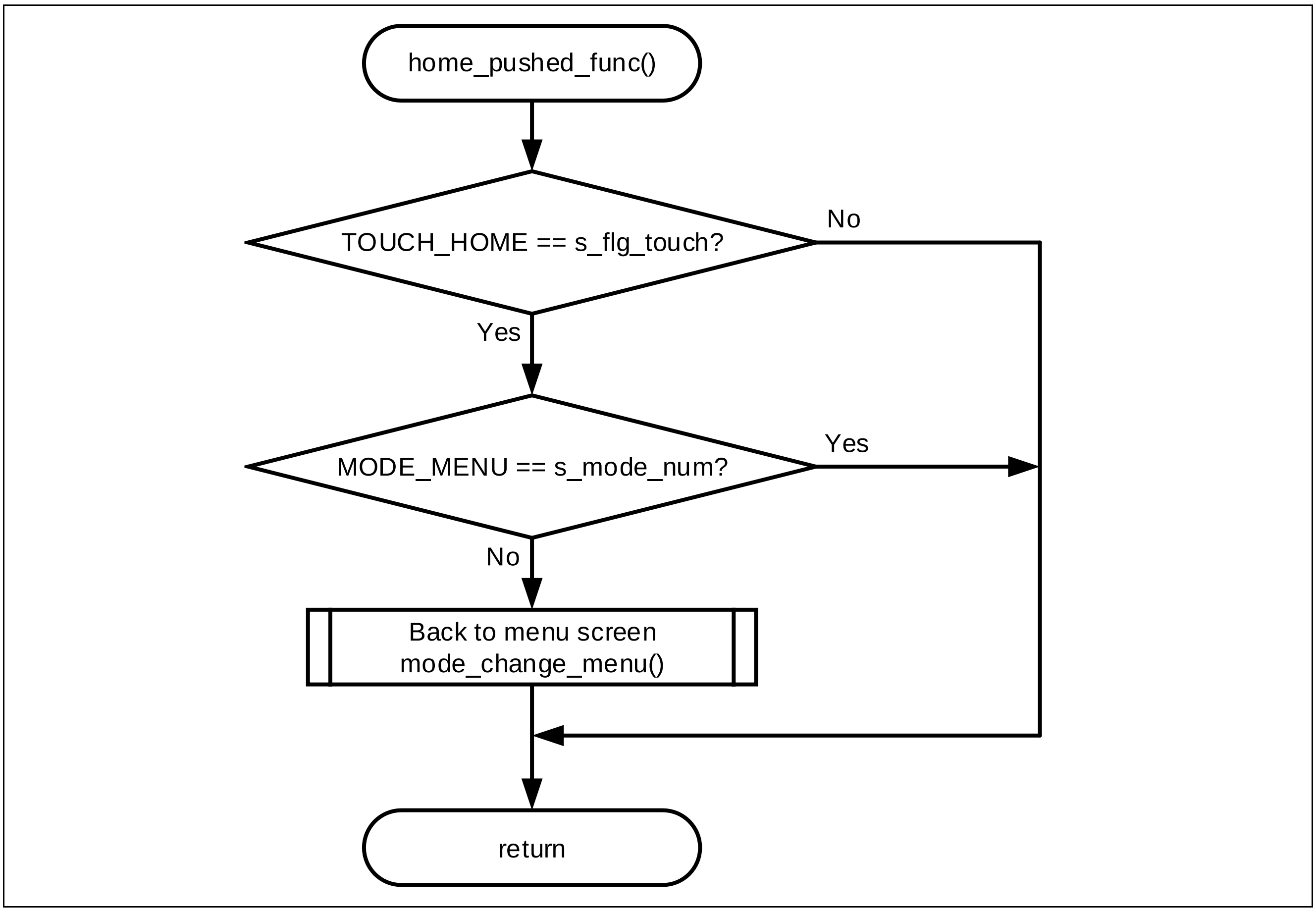


**Figure 3-7 Flowchart when the "home" button is touched**

### 3.2.5 Processing when the "select" button is touched

The flowchart when the "select" button is touched is shown below.

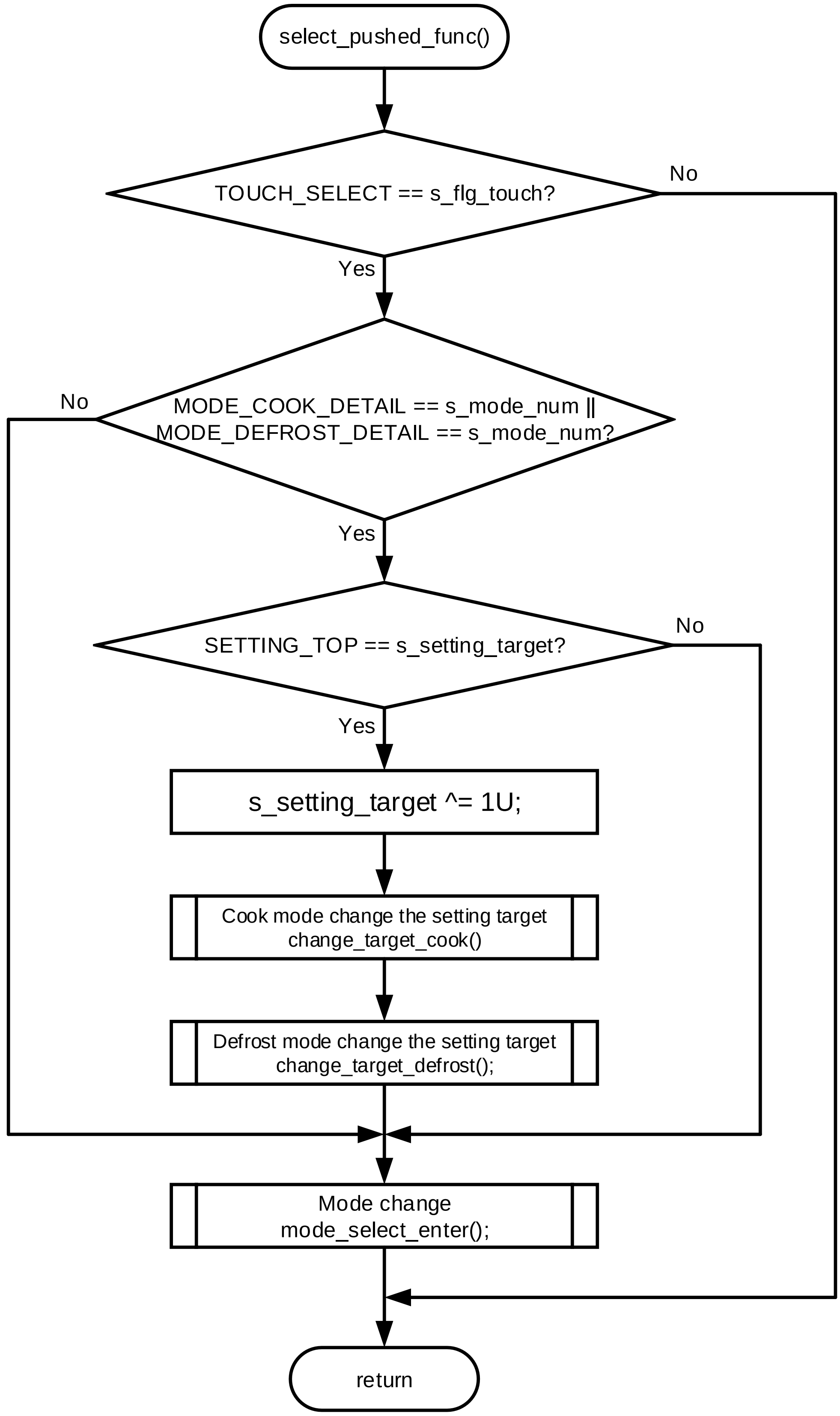


**Figure 3-8 Flowchart when the "select" button touch is touched**

### 3.2.6 Processing when the "start" button is touched

The flowchart when the "start" button is touched is shown below.

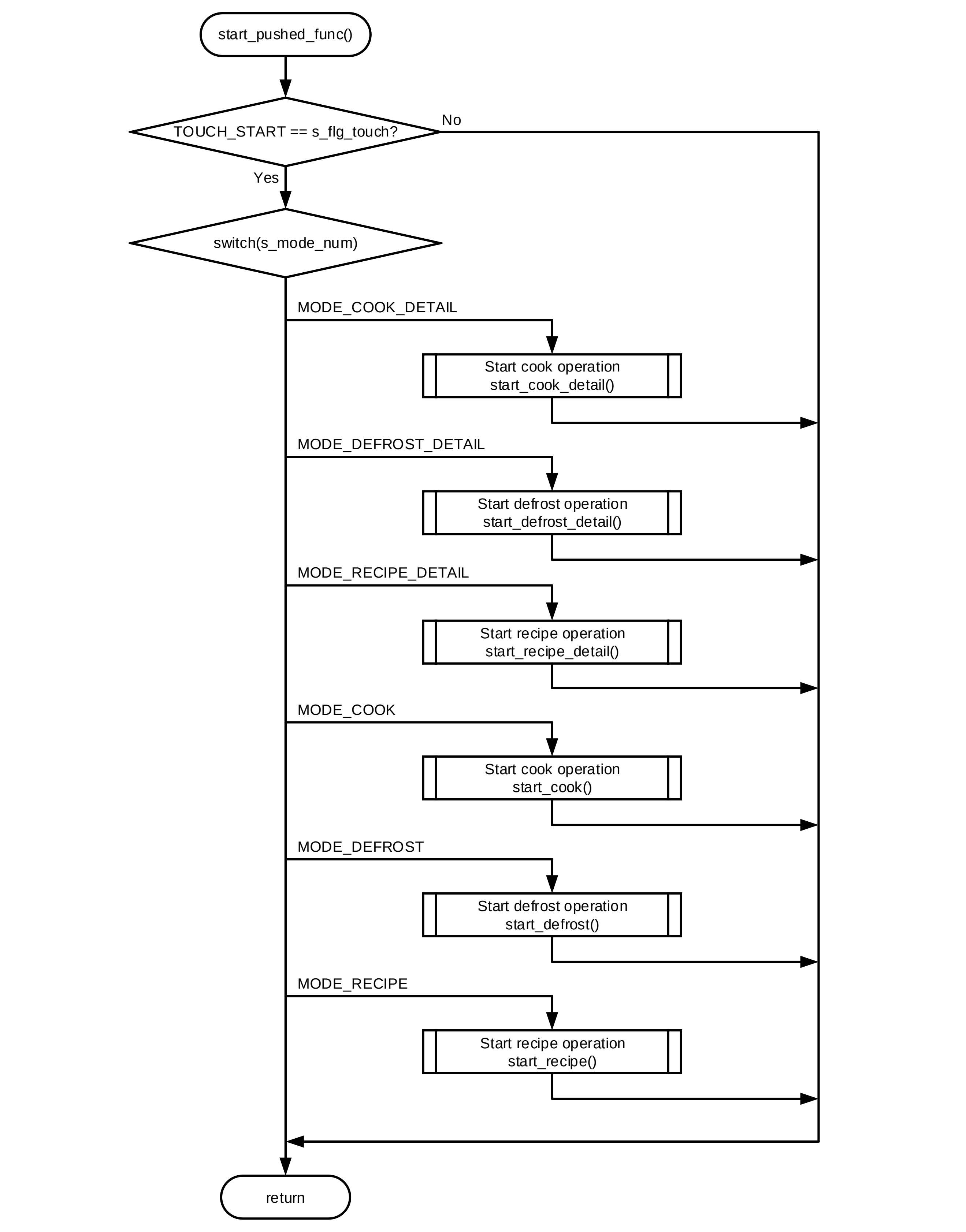


**Figure 3-9 Flowchart when the "select" button is touched**

### 3.2.7 Processing during cooking

The flowchart during cooking is shown below.

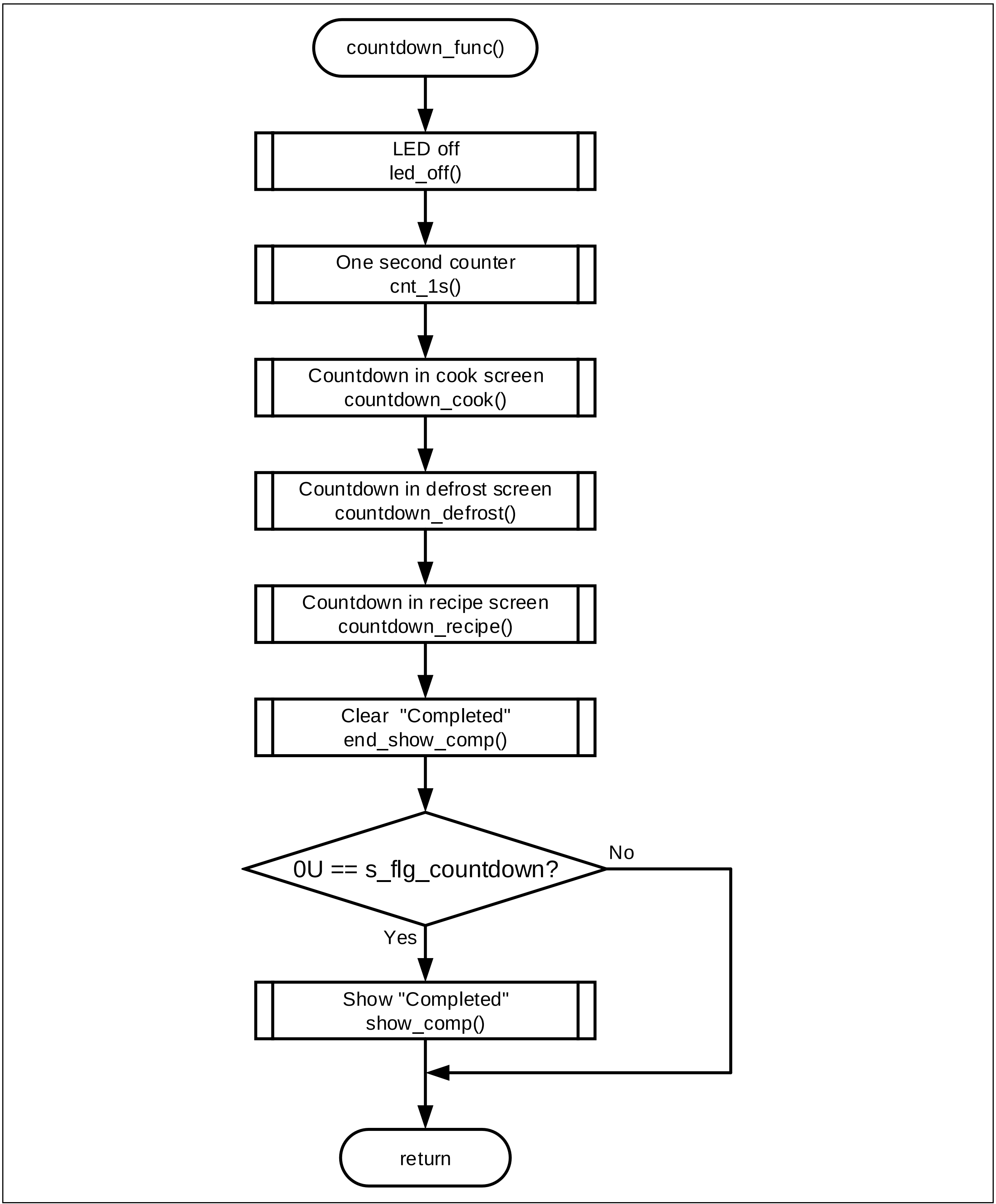


**Figure 3-10 Flowchart during cooking**

### 3.2.8 Overall flowchart of touch control

The overall flowchart of touch control is shown below.

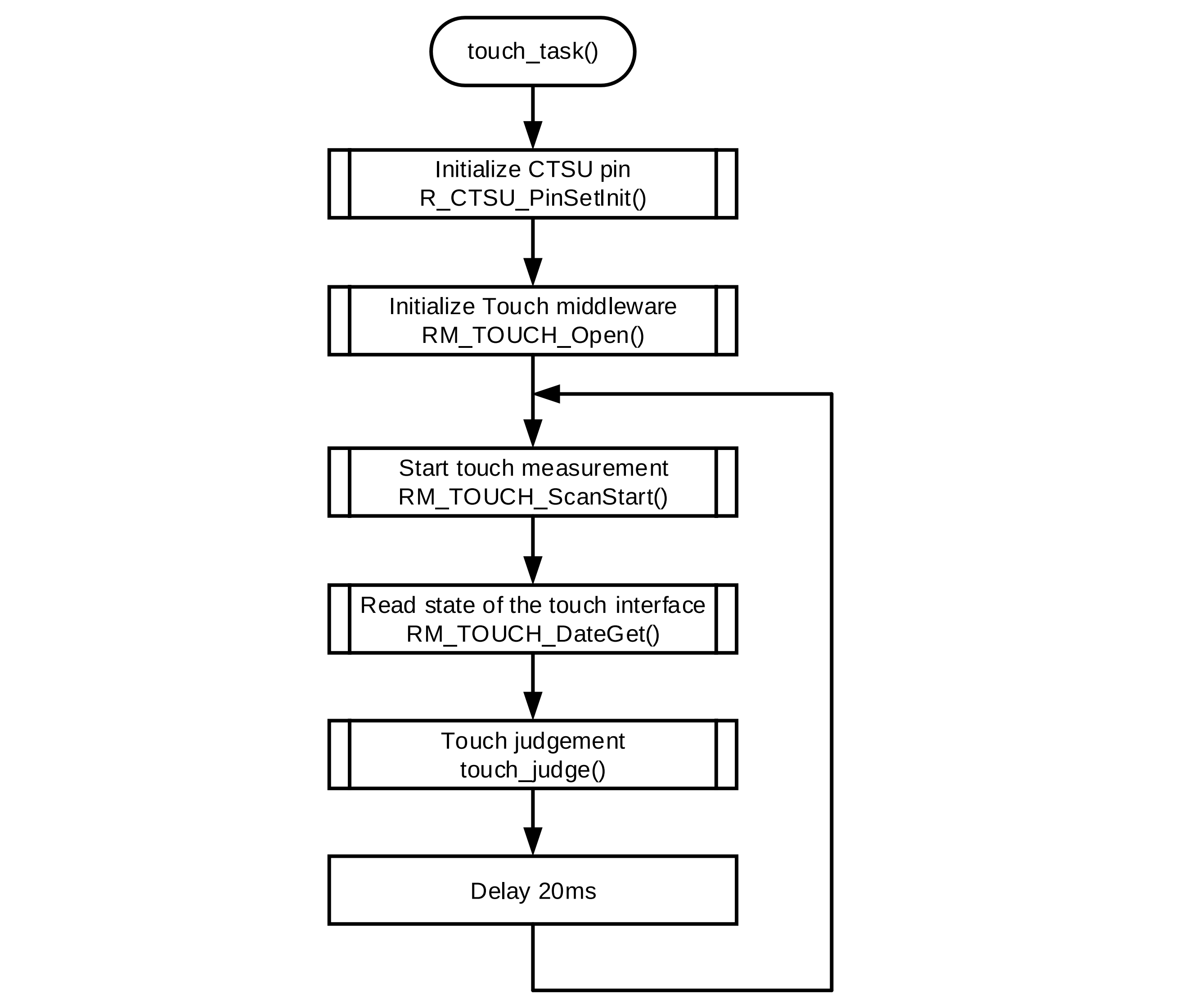


**Figure 3-11 Overall flowchart of touch control**

### 3.2.9 Processing of touch judgement

The flowchart of touch judgement is shown below.

If the left side of the touch slider is touched after touching the right side of the touch slider, the touch slider is judged to have slid to the left. The same is true on the opposite side.

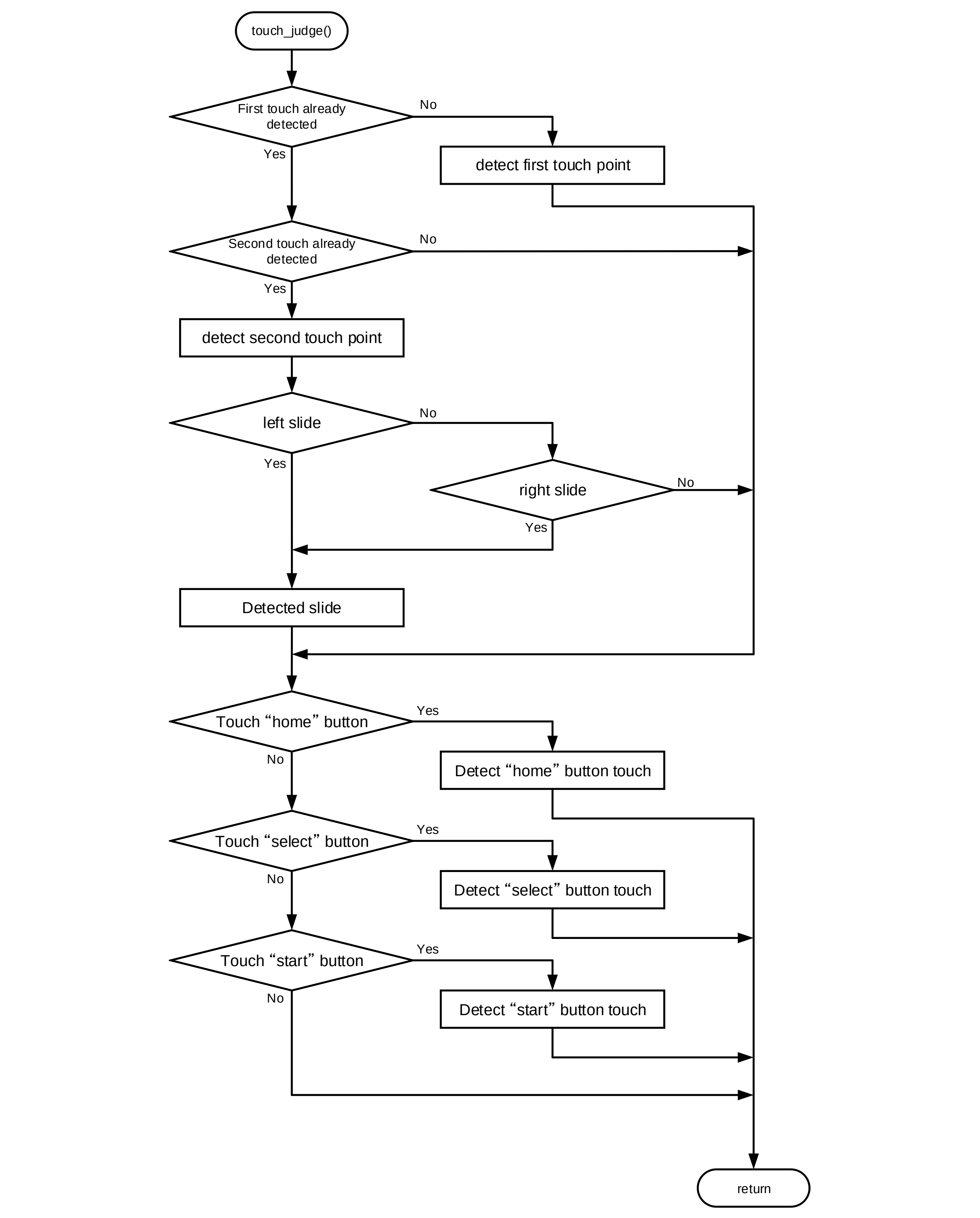


**Figure 3-12 Flowchart of touch judgement**

### 3.2.10 Processing of startup screen display

The flowchart of startup screen display is shown below.

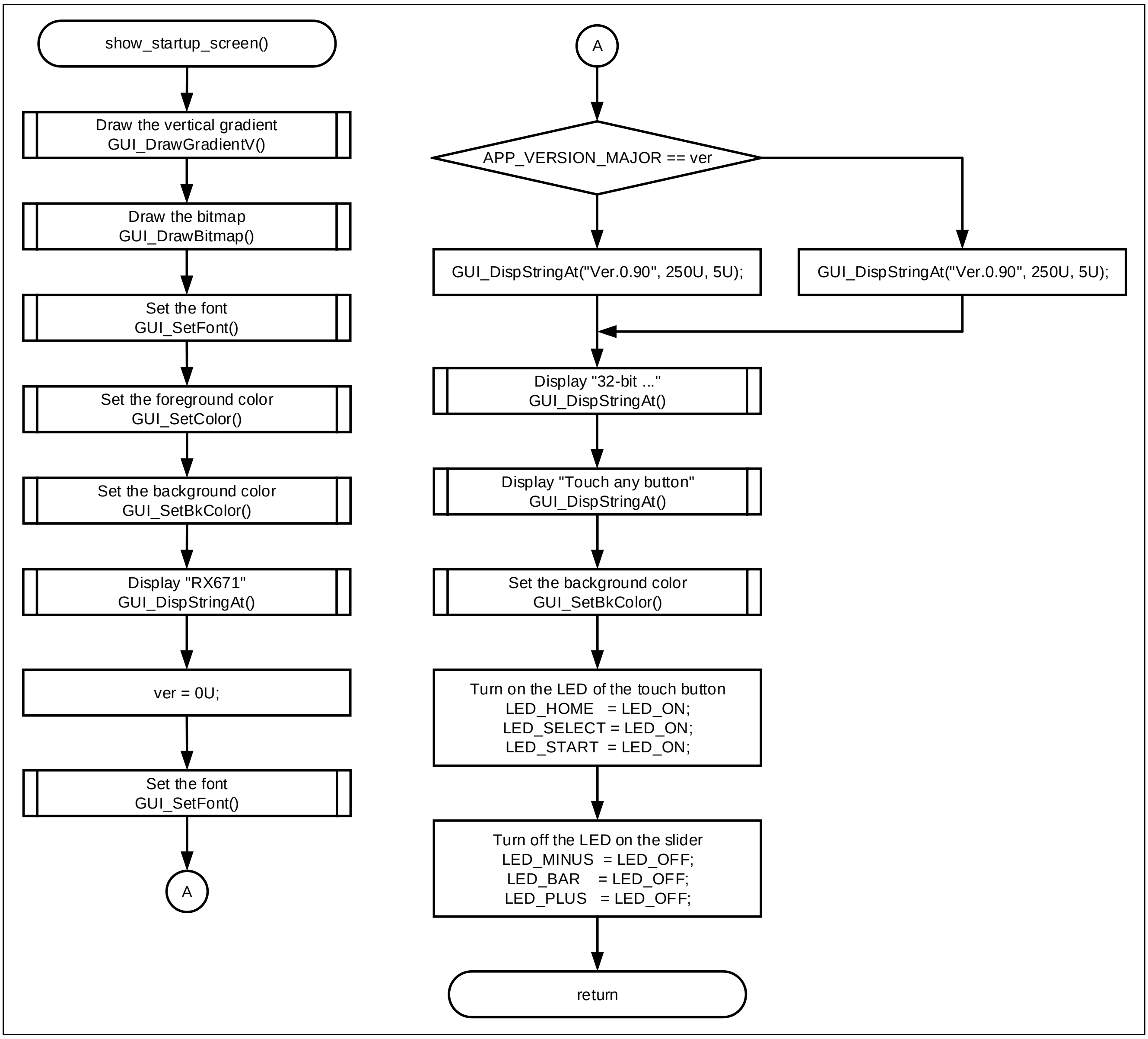


**Figure 3-13 Flowchart of startup screen display**

### 3.2.11 Processing of 5 seconds wait

The flowchart of 5 seconds wait is shown below.

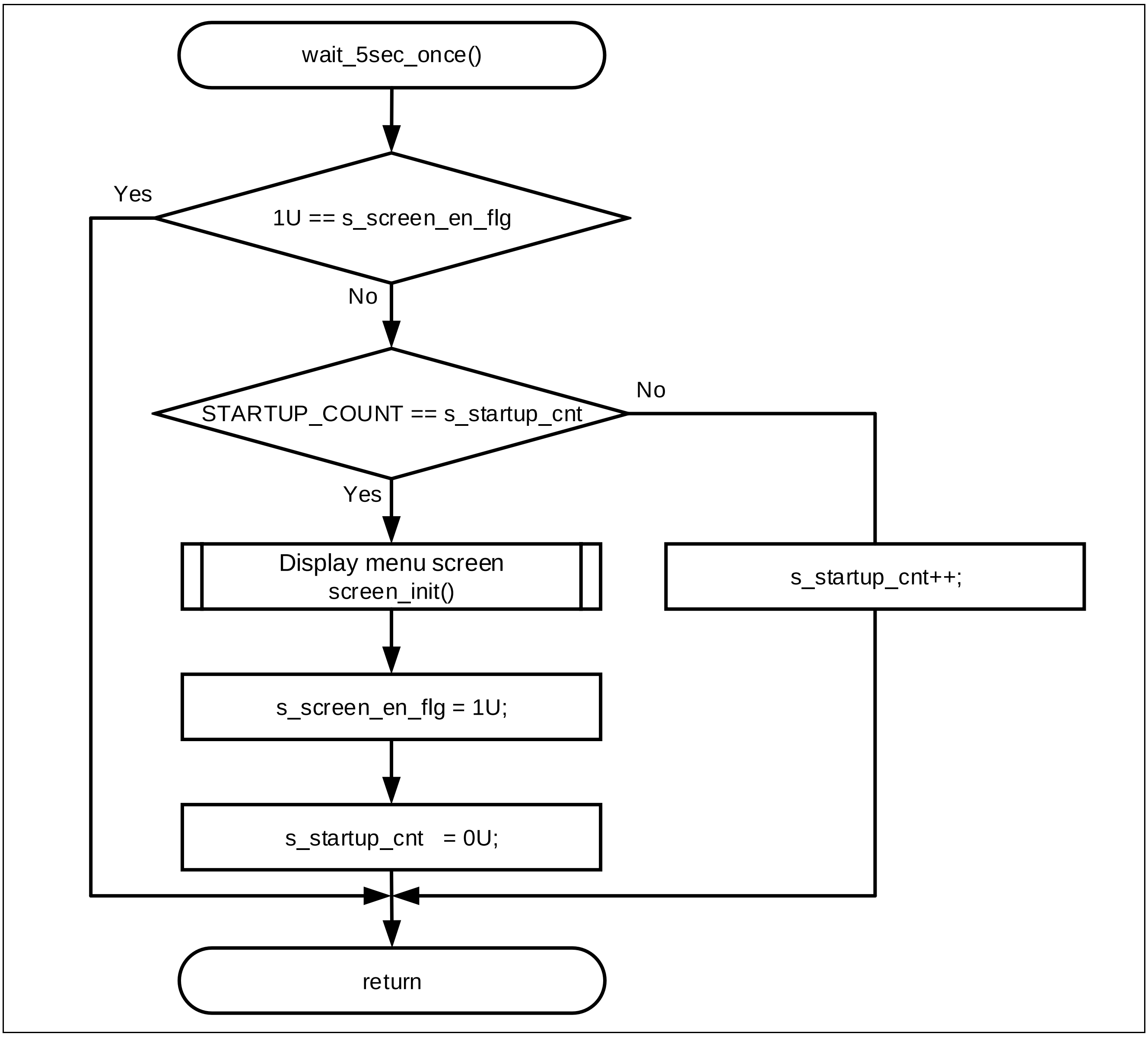


**Figure 3-14 Flowchart of 5 seconds wait**

### 3.2.12 Processing of flag clear for touch

The flowchart of flag clear for touch is shown below.

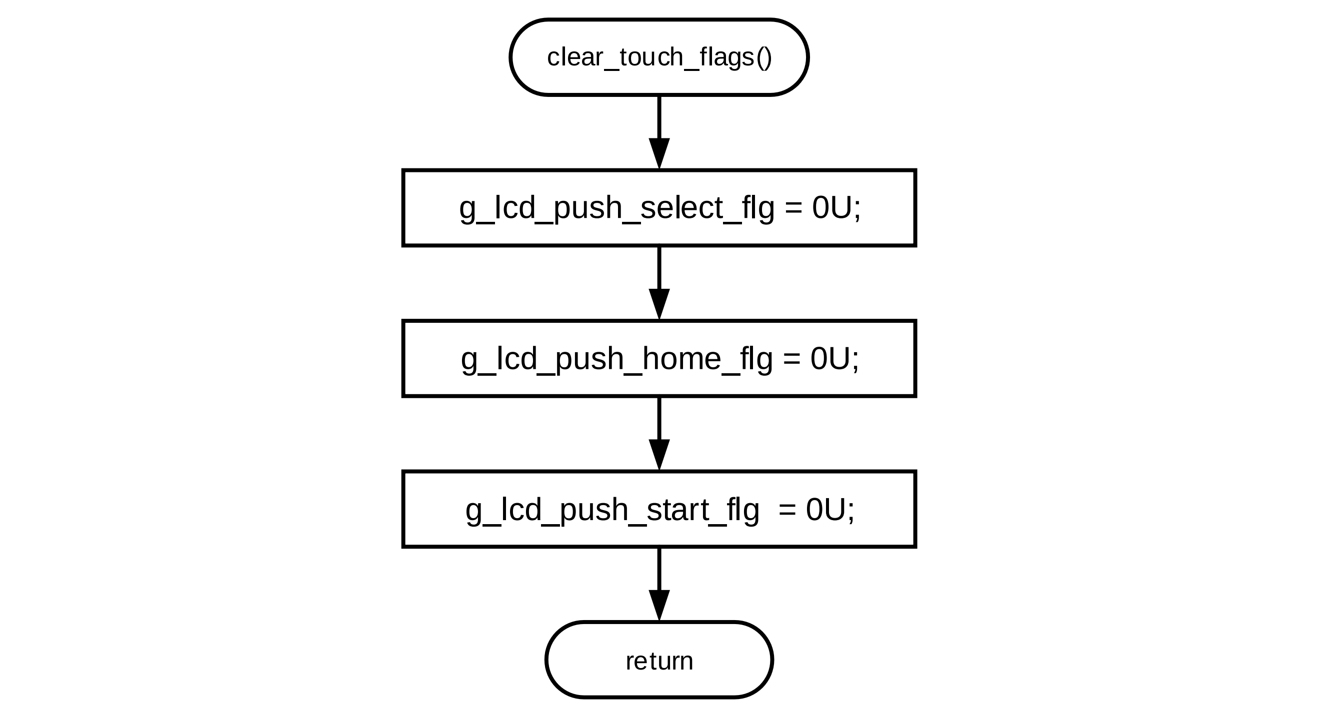


**Figure 3-15 Flowchart of flag clear touch**

### 3.2.13 Processing of screen initialization

The flowchart of screen initialization is shown below.

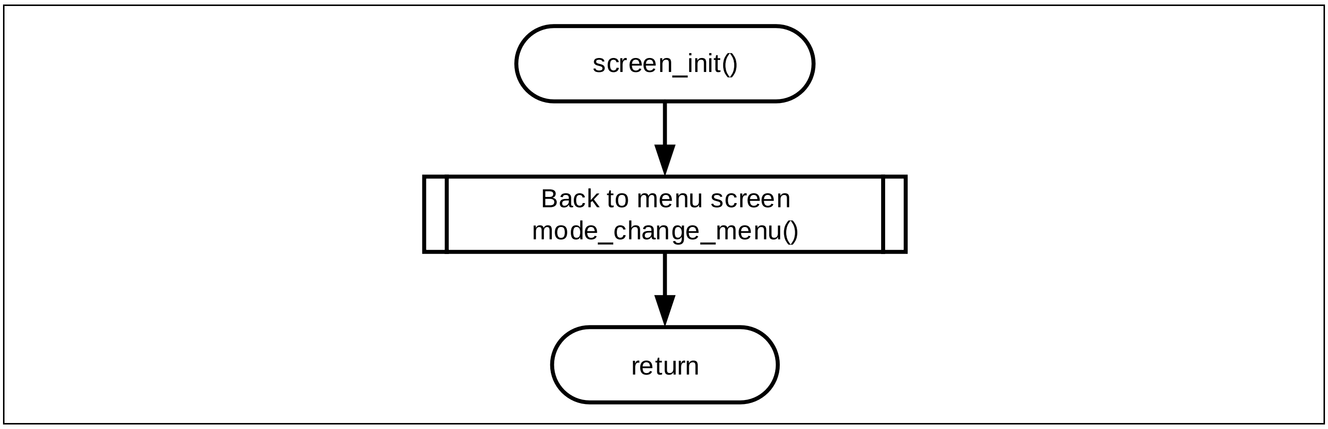


**Figure 3-16 Flowchart of screen initialization**

### 3.2.14 Processing at moving to the menu screen

The flowchart for moving to the menu screen is shown below.

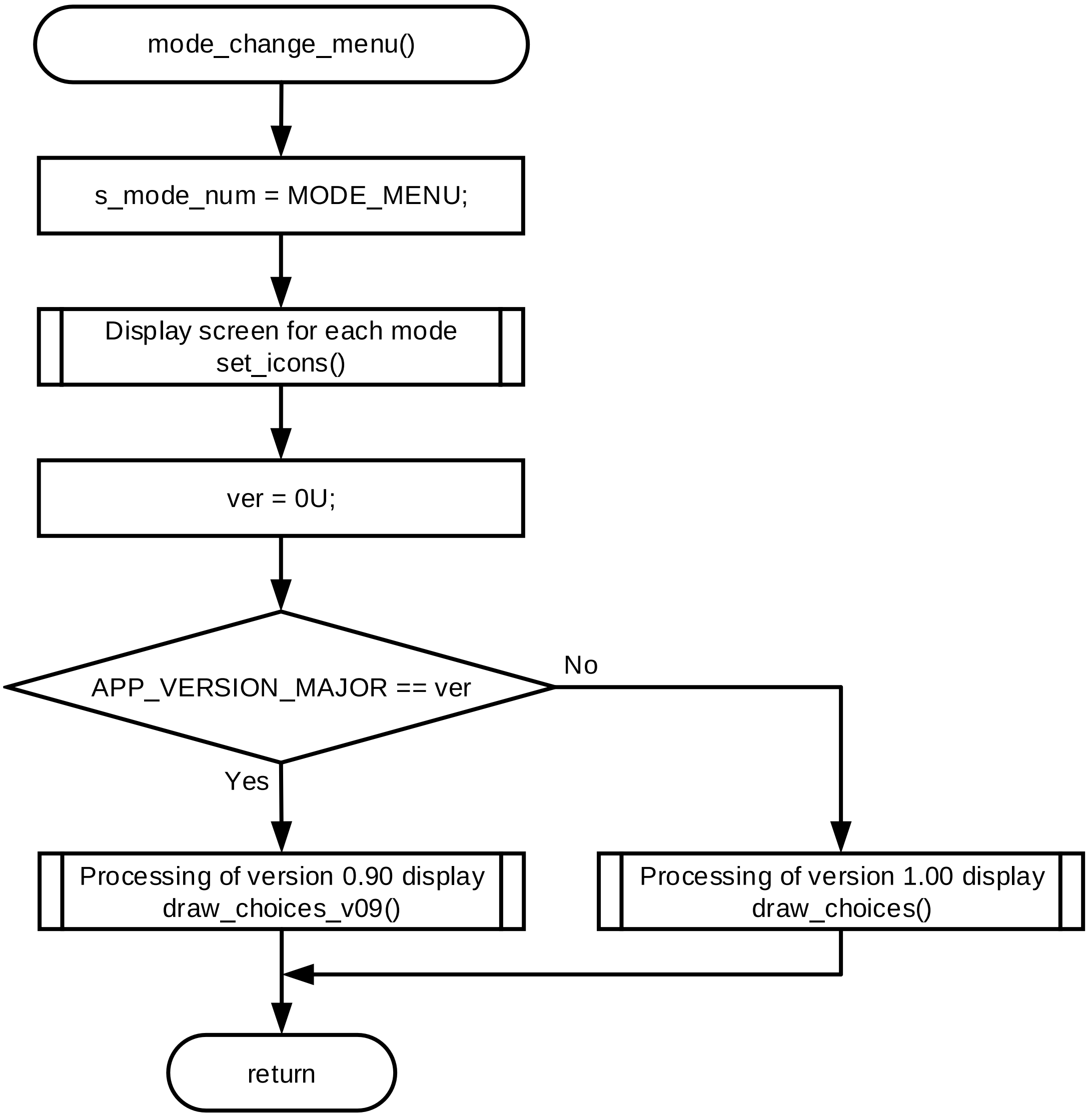


**Figure 3-17 Flowchart for moving to the menu screen**

### 3.2.15 Processing of setting LED pattern to sleep

The flowchart of setting LED pattern to sleep is shown below.

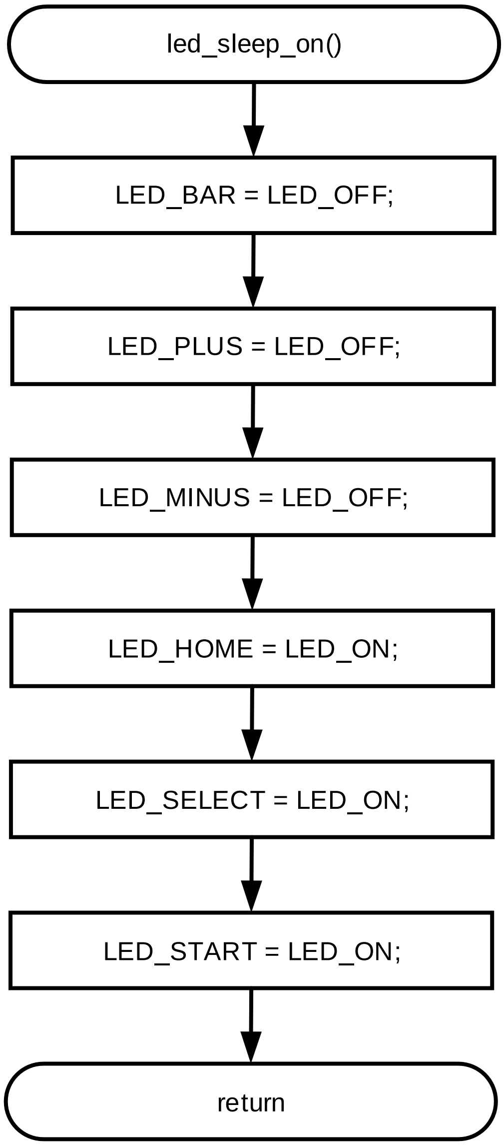


**Figure 3-18 Flowchart of setting LED pattern to sleep**

## 3.3 Pins Used

The following shows lists pins used in this sample program.

**Table 3-1 List of Pins and Functions**

| Pin name | Input/Output | Function |
|---|---|---|
| P12/RXD2 | Input | UART2 receiving pin |
| P13/TXD2 | Output | UART2 sending pin |
| PE1/TXD12 | Output | UART12 sending pin |
| PE2/RXD12 | Input | UART12 receiving pin |
| PA6/CTS12 | Input | CTS signal input pin |
| P16/RXD3 | Input | USB-UART3 receiving pin |
| P17/TXD3 | Output | USB-UART3 sending pin |
| P27/RSPCKB-A | Input/Output | RSPI1 clock pin |
| PE7/MISOB-B | Input | RSPI1 MISO pin |
| PE6/MOSIB-B | Output | RSPI1 MOSI pin |
| PC4/TSCAP | - | TSCAP pin |
| P34/TS0 | Input | Electrostatic capacitance measurement pin |
| P26/TS3 | Input | Electrostatic capacitance measurement pin |
| P53/TS12 | Input | Electrostatic capacitance measurement pin |
| PC6/TS13 | Input | Electrostatic capacitance measurement pin |
| PC5/TS14 | Input | Electrostatic capacitance measurement pin |
| PC1/TS15 | Input | Electrostatic capacitance measurement pin |
| PC0/TS16 | Input | Electrostatic capacitance measurement pin |
| PD40~43, PD2, PD3 | Output | LED pin |
| PB6/TIOCA5 | Output | Buzzer pin |

## 3.4 Sample Program Structure

### 3.4.1 Peripheral Functions Used

The following shows lists peripheral functions used in this sample program.

**Table 3-2 List of Peripheral Functions Used and Functions**

| Peripheral Functions | Functions |
|---|---|
| RSPI1 | SPI Communication with LCD module |
| DMAC | Used for RAM to RSPI transfer |
| SCI2, SCI12 | UART communication with Wi-Fi module (2 systems) |
| S12AD0 | Used for inside CTSU FIT |
| CTSU | Used for touch buttons and touch slider |
| CMT0 | Used for internal emWin FIT |
| CMT2 | Used for RTOS |
| PORT | Used for LED |
| TPU5 | Used for buzzer |

### 3.4.2 Components Used

The following shows lists components used in this sample program.

**Table 3-3 List of Components Used and Functions**

| Components | Abbreviation | Version |
|---|---|---|
| ADC Driver | r_s12ad_rx | 5.00 |
| AWS_device_shadow | AWS_device_shadow | 1.0.110 |
| AWS_ggd | AWS_ggd | 1.0.110 |
| AWS_mqtt | AWS_mqtt | 1.0.110 |
| AWS_secure_socket | AWS_secure_socket | 1.0.110 |
| AWS_tcp_ip | AWS_tcp_ip | 1.0.110 |
| Board Support Package | r_bsp | 7.21 |
| Byte-based circular buffer library | r_byteq | 2.10 |
| CMT driver | r_cmt_rx | 5.40 |
| CTSU QE API | r_ctsu_qe | 2.20 |
| DMAC driver | r_dmaca_rx | 3.00 |
| Flash API for RX100, RX200, RX600, and RX700 | r_flash_rx | 4.91 |
| FreeRTOS_kernel | FreeRTOS_kernel | 1.0.110 |
| FreeRTOS_Object | FreeRTOS_Object | 1.0.112 |
| GPIO Driver | r_gpio_rx | 4.70 |
| Graphic Library with Graphical User Interface | r_emwin_rx | 6.32.a.1.00 |
| PWM Mode Timer | Config_TPU5 | 1.12.0 |
| RSPI Driver | r_rspi_rx | 3.04 |
| SCI Driver | r_sci_rx | 4.60 |
| Touch QE API | rm_touch_qe | 2.20 |
| Wi-Fi Module control functions for Renesas MCUs | r_wifi_sx_ulpgn | 1.16 |
| Port | Config_PORT | 2.4.1 |

### 3.4.3 Peripheral Function Settings

The Smart Configurator settings used in this sample program are shown below. The items and settings in each table in the Smart Configurator settings are described in the notation on the configuration screen.

Settings not listed are assumed to be default settings.

**Table 3-4 Parameters of Smart Configurator (1/6)**

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Clock | | The following settings are made on the "Clocks" Tab. |
| | VCC | 3.3 (V) |
| | Main clock | Stopped: Unchecked. |
| | PLL circuit setting | Frequency Division: ×1<br>Frequency Multiplication: ×15 |
| | HOCO clock | Operation: Checked.<br>HOCO oscillation enabled after reset |
| | LOCO clock | Stopped: Unchecked. |
| | IWDT dedicated clock | Clock source: HOCO<br>Flash IF clock (FCLK): 60MHz<br>System clock (ICLK): 120MHz<br>Peripheral module clock (PCLKA): 120MHz<br>Peripheral module clock (PCLKB): 60MHz<br>Peripheral module clock (PCLKC): 60MHz<br>Peripheral module clock (PCLKD): 60MHz |
| Smart Configurator >> System | | Debugging interfaces setting: FINE |
| Smart Configurator >> Components >> r_bsp | | Default settings except following changes |
| | Heap size | 0x4000 |
| | ROM Cache Enable Register | Disabled |
| | Software Interrupt Unit2 (SWINT2) | Used |
| | Software Interrupt Task Buffer Number | 8 |
| | Initial value of the software interrupt priority | Priority level 1 |
| | Serial terminal select | Enabled |
| | Channel for serial terminal | Channel 3 |
| | Bitrate for serial terminal | 115200 |
| | Interrupt priority serial terminal | Priority level 3 |
| | HOCO Trimming select | Disabled |
| Smart Configurator >> Components >> r_dmaca_rx | | Default settings are used. |
| Smart Configurator >> Components >> r_s12ad_rx | | Default settings except following changes |
| | Resources >> S12AD | |
| | S12AD0 | Checked. |
| Smart Configurator >> Components >> r_ctsu_qe | | Default settings except following changes |
| | TSCAP pin | Use: Checked. |
| | TS0 pin | Use: Checked. |
| | TS2 pin | Use: Checked. |
| | TS3 pin | Use: Checked. |
| | TS12 pin | Use: Checked. |
| | TS13 pin | Use: Checked. |
| | TS14 pin | Use: Checked. |
| | TS15 pin | Use: Checked |
| | TS16 pin | Use: Checked. |

Table 3-5 Parameters of Smart Configurator (2/6)

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Components >> r_gpio_rx | | Default settings are used. |
| Smart Configurator >> Components >> r_flash_rx | | Default settings except following changes |
| | Enable code flash programming (FLASH_CFG_CODE_FLASH_ENABLE) | Includes code to program ROM area |
| | Enable BGO/Non-blocking data flash operations (FLASH_CFG_DATA_FLASH_BGO) | Enable BGO mode |
| | Enable BGO/Non-blocking code flash operations (FLASH_CFG_CODE_FLASH_BGO) | Enable BGO mode |
| | Enable code flash self-programming (FLASH_CFG_CODE_FLASH_RUN_FROM_ROM) | Programming code flash while executing from another segment in ROM |
| Smart Configurator >> Components >> r_rspi_rx | | Default settings except following changes |
| | Dummy data of reception | 0x00 |
| | RSPI channel 0 | Unused |
| | RSPI channel 1 | Used |
| | RSPI channel 2 | Unused |
| | Interrupt priority level of RSPI channel 1 | Level 3 |
| | RSPI1 | Checked |
| | RSPCKB pin | Use: Checked. |
| | MOSIB pin | Use: Checked. |
| | MISOB pin | Use: Checked. |
| Smart Configurator >> Components >> r_sci_rx | | Default settings except following changes |
| | Use ASYNC mode | Include |
| | Use SSPI mode | Include |
| | Include software support for channel 2 (SCI_CFG_CH2_INCLUDED) | Include |
| | Include software support for channel 3 (SCI_CFG_CH3_INCLUDED) | Include |
| | Include software support for channel 12 (SCI_CFG_CH12_INCLUDED) | Include |
| | ASYNC mode TX queue buffer size for channel 3 (SCI_CFG_CH3_TX_BUFSIZ) | 2048 |
| | ASYNC mode TX queue buffer size for channel 12 (SCI_CFG_CH12_TX_BUFSIZ) | 2048 |
| | ASYNC mode RX queue buffer size for channel 3 (SCI_CFG_CH3_RX_BUFSIZ) | 2048 |
| | ASYNC mode RX queue buffer size for channel 12 (SCI_CFG_CH12_RX_BUFSIZ) | 2048 |
| | Resources >> SCI | |
| | SCI2 | Checked. |
| | RXD2/SMISO2/SSCL2 Pin | Use: Checked. |
| | TXD2/SMOSI2/SSDA2 Pin | Use: Checked. |
| | SCI3 | Checked |
| | RXD3/SMISO3/SSCL3 Pin | Use: Checked |
| | TXD3/SMOSI3/SSDA3 Pin | Use: Checked |
| | SCI12 | Checked. |
| | RXD12/SMISO12/SSCL12 Pin | Use: Checked. |
| | TXD12/SMOSI12/SSDA12 Pin | Use: Checked. |
| | CTS12#/RTS12#/SS12# Pin | Use: Checked. |

**Table 3-6 Parameters of Smart Configurator (3/6)**

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Components >> r_cmt_rx | | Default settings are used. |
| Smart Configurator >> Components >> rm_touch_qe | | Default settings are used. |
| Smart Configurator >> Components >> r_byteq | | Default settings are used. |
| Smart Configurator >> Components >> r_wifi_sx_ulpgn | | Default settings except following changes |
| | SCI Channel number for SX-ULPGN Initial Command Port for AT command communication (WIFI_CFG_SCI_CHANNEL) | 12 |
| | SCI Channel number for SX-ULPGN Second Command Port for AT command communication (WIFI_CFG_SCI_SECOND_CHANNEL) | 2 |
| | General-purpose port PDR register connected to the SX-ULPGN EN pin (WIFI_CFG_RESET_PORT) | PORTH |
| | Configure RTS Port No. for WIFI_CFG_SCI_CHANNEL (WIFI_CFG_RTS_PORT) | PORTH |
| | Configure RTS Pin No. for WIFI_CFG_SCI_CHANNEL (WIFI_CFG_RTS_PIN) | 1 |
| | Socket Receive buffer size (WIFI_CFG_SOCKETS_RECEIVE_BUFFER_SIZE) | 1024 |
| Smart Configurator >> Components >> r_emwin_rx | | Other than the changes listed below, default settings are used. |
| | Configurations >> BasicSetting | |
| | Work area size for GUI | 10000 |
| | Horizontal LCD size | 240 |
| | Vertical LCD size | 320 |
| | Color depth | 16 bit per pixel |
| | LCD orientation | ORIENTATION_CCW |
| | Configurations >> Select LCD Interface | |
| | LCD interface | LCD_IF_RSPI |
| | Configurations >> Select LCD Interface >> SPI Interface Setting | |
| | LCD interface channel number | 1 |
| | Select LCD Driver IC | LCD_DRV_IC_ILI9341 |
| | Communication baud rate of LCD interface | 30000000 |
| | Use or unused display cache | Unuse: Unchecked. |
| | Configurations >> Select LCD Interface >> LCD Interface Pin Setting | |
| | Use Display Signal Pin | Use Display Signal Pin |
| | Display Signal Pin | GPIO_PORT_A_PIN_1 |
| | Use Backlight Pin | Use Backlight Pin |
| | Backlight Pin | GPIO_PORT_D_PIN_6 |
| | Use Data/Command Pin | Use Data/Command Pin |
| | Data/Command Pin | GPIO_PORT_A_PIN_2 |
| | Use Chip Select Pin | Use Chip Select Pin |
| | Chip Select Pin | GPIO_PORT_D_PIN_5 |
| | Configurations >> Select Touch Interface | |
| | Use Touch function | Not use Touch function: Unchecked. |

**Table 3-7 Parameters of Smart Configurator (4/6)**

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Components >> FreeRTOS_Kernel | | Other than the changes listed below, default settings are used. |
| | The total amount of RAM available in the FreeRTOS heap | (size_t)(200U * 1024U) |
| | Tick vector | _CMT2_CMI2 |
| Smart Configurator >> Components >> FreeRTOS_Object | | Other than the changes listed below, default settings are used. |
| | Tasks | Initialize: kernel start<br>Task Code: touch_task<br>Task Name: touch_task<br>Stack Size: 512<br>Task Handler: NULL<br>Parameter: NULL<br>Priority: 1 |
| | | Initialize: kernel start<br>Task Code: emwin_task<br>Task Name: emwin_task<br>Stack Size: 512<br>Task Handler: NULL<br>Parameter: NULL<br>Priority: 1 |
| Smart Configurator >> Components >> AWS_device_shadow | | Default settings are used. |
| Smart Configurator >> Components >> AWS_ggd | | Default settings are used. |
| Smart Configurator >> Components >> AWS_mqtt | | Default settings are used. |
| Smart Configurator >> Components >> AWS_secure_soket | | Default settings are used. |
| Smart Configurator >> Components >> AWS_tcp_ip | | Default settings are used. |

**Table 3-8 Parameters of Smart Configurator (5/6)**

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Pins >> Serial Communication Interface >> SCI2 | | Uncheck all settings except the following |
| | RXD2 | Use: Checked.<br>Terminal Assignment: Set P12 |
| | TXD2 | Use: Checked.<br>Terminal Assignment: Set P13 |
| Smart Configurator >> Pins >> Serial Communication Interface >> SCI3 | | Uncheck all settings except the following |
| | RXD3 | Use: Checked.<br>Terminal Assignment: Set P16 |
| | TXD3 | Use: Checked.<br>Terminal Assignment: Set P17 |
| Smart Configurator >> Pins >> Serial Communication Interface >> SCI5 | | Uncheck all settings except the following |
| | SCK5 | Use: Checked.<br>Terminal Assignment: Set PA1 |
| | SMISO5 | Use: Checked.<br>Terminal Assignment: Set PA2 |
| | SMOSI5 | Use: Checked.<br>Terminal Assignment: Set PA4 |
| Smart Configurator >> Pins >> Serial Communication Interface >> RSPI1 | | Uncheck all settings except the following |
| | MISOB | Use: Checked.<br>Terminal Assignment: Set PE7 |
| | MOSIB | Use: Checked.<br>Terminal Assignment: Set PE6 |
| | RSPCKB | Use: Checked.<br>Terminal Assignment: Set P27 |

**Table 3-9 Parameters of Smart Configurator (6/6)**

| Category | Item | Setting/Description |
|---|---|---|
| Smart Configurator >> Components >> Config_PORT | | Other than the changes listed below, default settings are used. |
| | PORT4 | Checked. |
| | P40 | Output: Checked<br>Output 1: Checked |
| | P41 | Output: Checked<br>Output 1: Checked |
| | P42 | Output: Checked<br>Output 1: Checked |
| | P43 | Output: Checked<br>Output 1: Checked |
| | PORTD | Checked. |
| | PD2 | Output: Checked<br>Output 1: Checked |
| | PD3 | Output: Checked<br>Output 1: Checked |
| Smart Configurator >> Components >> Config_TPU5 | | Other than the changes listed below, default settings are used. |
| | Counter clear source | TGRB5 compare match |
| | Counter clock selection | PCLK/64, rising edge |
| | TIOCA5 pin | Output initial 0, 1 at compare match |
| | TGRB compare match operation | Output 0 from TIOCA5 pin |
| | PWM cycle | 504μs |
| | TGRA initial value | 38 |
| | TGRB initial value | 472 |

### 3.4.4 File Structure

The following shows file structure by sample program.

**Table 3-10 File Structure**

| Folder name, File name | Outline |
|---|---|
| application_code | - |
| ├ LCD | - |
| │ ├ Resource | Folders for images and fonts |
| │ └ Source | - |
| │ ├ LCD_custom_func.c | Source file for LCD related |
| │ └ LCD_custom_func.h | Header file for LCD related |
| ├ renesas_code | - |
| │ ├ frtos_startup | Generated folder for Amazon FreeRTOS |
| │ ├ utility | |
| │ ├ main_task.c | Amazon FreeRTOS main task |
| │ └ frtos_skeleton | - |
| │ ├ emwin_task.c | Task for emWin control |
| │ ├ task_function.h | Include file for emwin_task.c, touch_task.c |
| │ └ touch_task.c | Task for touch control |
| ├ touch | - |
| │ ├ touch_func.c | Source file for touch related |
| │ └ touch_func.h | Header file for touch related |
| ├ qe_gen | Generated folder for QE for capacitive touch |
| └ main.c | Source file for main processing |
| config_files | Generated folder for Amazon FreeRTOS |
| demos | |
| freertos_kernel | |
| libraries | |
| QE-Touch | Generated folder for QE for capacitive touch |
| vendors | Generated folder for Amazon FreeRTOS |
| └ renesas | - |
| └ boards | - |
| └ rx671-rsk | - |
| └ aws_demos | - |
| └ src | - |
| └ smc_gen | Smart Configurator generation |
| ├ Config_PORT | |
| ├ Config_TPU5 | |
| ├ general | |
| ├ r_cmt_rx | |
| ├ r_config | |
| ├ r_ctsu_qe | |
| ├ r_dmaca_rx | |
| ├ r_emwin_rx | |
| ├ r_gpio_rx | |
| ├ r_pincfg | |
| ├ r_rspi_rx | |
| └ rm_touch_qe | |

### 3.4.5 Variables

The following shows the variables that are used in this sample program.

**Table 3-11 List of variables used in the sample code**

| Variable name | Type | Contents |
|---|---|---|
| g_sleep_flg | uint8_t | Flag indicating LCD sleep state |
| g_lcd_left_slide_flg | uint8_t | Flag indicating that touch slider is slid to the left |
| g_lcd_right_slide_flg | uint8_t | Flag indicating that touch slider is slid to the right |
| g_lcd_push_enter_flg | uint8_t | Flag indicating that "2" button was touched. |
| g_lcd_push_back_flg | uint8_t | Flag indicating that "1" button was touched. |
| g_screen_en_flg | uint8_t | Flag indicating initial screen status |
| s_flg_countdown | uint8_t | The countdown is in progress on the LCD flag |
| s_flg_touch | uint8_t | Touch buttons status |
| s_startup_cnt | uint8_t | Counter for initial screen display time management |
| s_sleep_cnt | uint16_t | Counter for no-operation time management |
| s_mode_num | uint8_t | Mode status |
| s_setting_target | uint8_t | Flags indicating screen status |

### 3.4.6 Constants

The following shows the constants that are used in this sample program.

**Table 3-12 List of constants used in the sample code**

| Constant Name | Setting Value | Contents |
|---|---|---|
| LCD_BACKLIGHT | (PORT7.PODR.BIT.B1) | Pin to control LCD backlight |
| OFF | (0U) | Value at backlight off |
| ON | (1U) | Value at backlight on |
| TOUCH_NO | (0U) | Value at no-operation |
| TOUCH_LEFT_SLIDE | (4U) | Value indicating that touch slider is slid to the left |
| TOUCH_RIGHT_SLIDE | (3U) | Value indicating that touch slider is slid to the right |
| TOUCH_SELECT | (1U) | Value indicating that "2" button was touched |
| TOUCH_HOME | (2U) | Value at moving to the previous screen |
| SLEEP_COUNT | (SLEEP_TIME / DELAY_TIME) | When the counter in the program equals this value, the LCD is turned off |
| MODE_RECIPE_DETAIL | (6U) | Value of detail setting in Recipe mode |
| MODE_COOK_DETAIL | (3U) | Value of detail setting in Cook mode |
| MODE_DEFROST_DETAIL | (4U) | Value of detail setting in Defrost mode |
| MODE_COOK | (1U) | Value of start cooking in Cook mode |
| MODE_DEFROST | (2U) | Value of start cooking in Defrost mode |
| MODE_RECIPE | (5U) | Value of start cooking in Recipe mode |
| TOUCH_START | (5U) | Value of execution in each mode |
| SETTING_TOP | (0U) | Value of initial screen |
| MODE_MENU | (0U) | Value of mode not selected |
| LED_HOME | (PORT4.PODR.BIT.B0) | P40 |
| LED_SELECT | (PORT4.PODR.BIT.B1) | P41 |
| LED_START | (PORT4.PODR.BIT.B2) | P42 |
| LED_MINUS | (PORT4.PODR.BIT.B3) | P43 |
| LED_BAR | (PORTD.PODR.BIT.B3) | PD3 |
| LED_PLUS | (PORTD.PODR.BIT.B2) | PD2 |
| LED_ON | (1U) | Value of LED turning on |
| LED_OFF | (0U) | Value of LED turning off |
| APP_VERSION_MAJOR | (1U) | Version display |
| STARTUP_COUNT | (STARTUP_TIME / DELAY_TIME) | Display the initial screen until the counter in the program equals this value |

### 3.4.7 Functions

The following shows the functions that are used in this sample program.

**Table 3-13 List of functions used in the sample code**

| Function name | Outline |
|---|---|
| emwin_task | LCD Control |
| GUI_Init | Initializing emWin |
| screen_init | Menu screen is displayed on LCD |
| change_screen | LCD screen update |
| slide_func | Processing at touch slider operation |
| select_pushed_func | Processing when "select" button is touched |
| home_pushed_func | Processing when "home" button is touched |
| start_pushed_func | Processing when "start" button is touched |
| countdown_func | Processing the countdown |
| slide_icons | Processing of cursor movement on menu screens and mode selection screens for Cook, Defrost and Recipe |
| setting_cook | Setting the number of watts and seconds in Cook mode |
| setting_defrost | Setting the level of defrosting and the number of grams in Defrost mode |
| setting_recipe | Setting the number of cupcakes in Recipe mode |
| mode_select_enter | Change the mode and display the LCD screen according to the mode |
| change_target_cook | Change the setting target of the detail setting screen in Cook mode |
| change_target_defrost | Change the setting target of the detail setting screen in Defrost mode |
| mode_change_menu | Move to the menu screen |
| start_cook | Start cooking in Cook mode |
| start_defrost | Start defrosting in Defrost mode |
| start_recipe | Start cooking in Recipe mode |
| countdown_cook | Processing during cooking in Cook mode |
| countdown_defrost | Processing during defrosting in Defrost mode |
| countdown_recipe | Processing during cooking in Recipe mode |
| touch_task | Initializes CTSU and calls the touch judgement function |
| touch_judge | Touch judgement |
| show_startup_screen | Processing of startup screen display |
| led_sleep_on | Set LED pattern to sleep |
| wait_5sec_once | Processing of 5 seconds wait |
| clear_touch_flags | Processing of flag clear for touch |
| draw_choices_v09 | Processing of version 0.90 display |
| draw_choices | Processing of version 1.00 display |
| cnt_1s | Processing of 1 second count |
| end_show_comp | Processing of cooking completion screen wait |
| show_comp | Processing of cooking completion screen display |
| start_cook_detail | Start of detail setting in Cook mode |
| start_defrost_detail | Start of detail setting in Defrost mode |
| start_recipe_detail | Start of detail setting in Recipe mode |
| led_off | Processing of LED turning off |

### 3.4.8 Function Specifications

The following shows function specifications that are used in this sample program.

[Function name] emwin_task

| | |
|---|---|
| **Outline** | LCD Control |
| **Header** | task_function.h |
| **Declaration** | void touch_task (void * pvParameters) |
| **Description** | Initializes emWin FIT and controls LCD. |
| **Arguments** | pvParameters |
| **Return value** | None |
| **Remarks** | None |

[Function name] GUI_Init

| | |
|---|---|
| **Outline** | Initializing emWin |
| **Header** | GUI.h |
| **Declaration** | void GUI_Init (void) |
| **Description** | Initializes emWin's internal data structures and variables. |
| **Arguments** | None |
| **Return value** | None |
| **Remarks** | None |

[Function name] screen_init

| | |
|---|---|
| **Outline** | Menu screen is displayed on LCD |
| **Header** | LCD_custom_func.h |
| **Declaration** | void screen_init (void) |
| **Description** | Menu screen is displayed on LCD. |
| **Arguments** | None |
| **Return value** | None |
| **Remarks** | None |

[Function name] change_screen

| | |
|---|---|
| **Outline** | LCD screen update |
| **Header** | LCD_custom_func.h |
| **Declaration** | void change_screen (void) |
| **Description** | Updates the LCD screen by touch operation. |
| **Arguments** | None |
| **Return value** | None |
| **Remarks** | None |

[Function name] show_startup_screen

| | |
|---|---|
| **Outline** | Processing of startup screen display |
| **Header** | LCD_custom_func.h |
| **Declaration** | void shoe_startup_screen (void) |
| **Description** | Performs startup screen display. |
| **Arguments** | None |
| **Return value** | None |
| **Remarks** | None |

[Function name] led_sleep_on

| | |
|---|---|
| **Outline** | Set LED pattern to sleep |
| **Header** | LCD_custom_func.h |
| **Declaration** | Void led_sleep_on (void) |
| **Description** | Sets LED pattern to sleep |
| **Arguments** | None |
| **Return value** | None |
| **Remarks** | None |

[Function name] touch_task

| | |
|---|---|
| **Outline** | Initializes CTSU and calls the touch judgement function |
| **Header** | task_function.h |
| **Declaration** | void touch_task (void * pvParameters) |
| **Description** | Initializes CTSU and calls the touch judgement function. |
| **Arguments** | pvParameters |
| **Return value** | None |
| **Remarks** | None |

[Function name] touch_judge

| | |
|---|---|
| **Outline** | Touch judgement |
| **Header** | touch_func.h |
| **Declaration** | void touch_judge (uint64_t button_status, uint16_t slider_position) |
| **Description** | Performs touch judgement and sets the judgement result to a flag. |
| **Arguments** | button_status, slider_position |
| **Return value** | None |
| **Remarks** | None |

### 3.4.9 ROM/RAM usage

ROM/RAM usage for this sample program is shown below.

**Table 3-14 ROM usage**

| Size(KByte) | Description |
|---|---|
| 550 | Amazon FreeRTOS |
| 200 | LCD Graphic data |
| 115 | emWin, LCD control |
| 15 | demo program, LCD_custom_func |
| 24 | Other |
| Total 904KByte | MAX 1024KByte (88% Used) : 1024KByte x 2Bank |

**Table 3-15 RAM usage**

| Size(KByte) | Description |
|---|---|
| 200 | OS Heap area |
| 16 | Heap area |
| 64 | AWS demo program |
| 37 | AWS cloud (exp. OTA) |
| 11 | emWin |
| 18 | Other |
| Total 356KByte | MAX 384KByte (92.7% Used) |

# 4. Importing a Project

The sample programs are distributed in e² studio project format. This section shows how to import a project into e² studio or CS+. After importing a project, check the build and debug settings.

## 4.1 Procedure in e² studio

To use sample programs in e² studio, follow the steps below to import them into e² studio. In projects managed by e² studio, do not use space codes, multibyte characters, and symbols such as "$", "#", "%" in folder names or paths to them.

(Note that depending on the version of e² studio you are using, the interface may appear somewhat different from the screenshots below.)

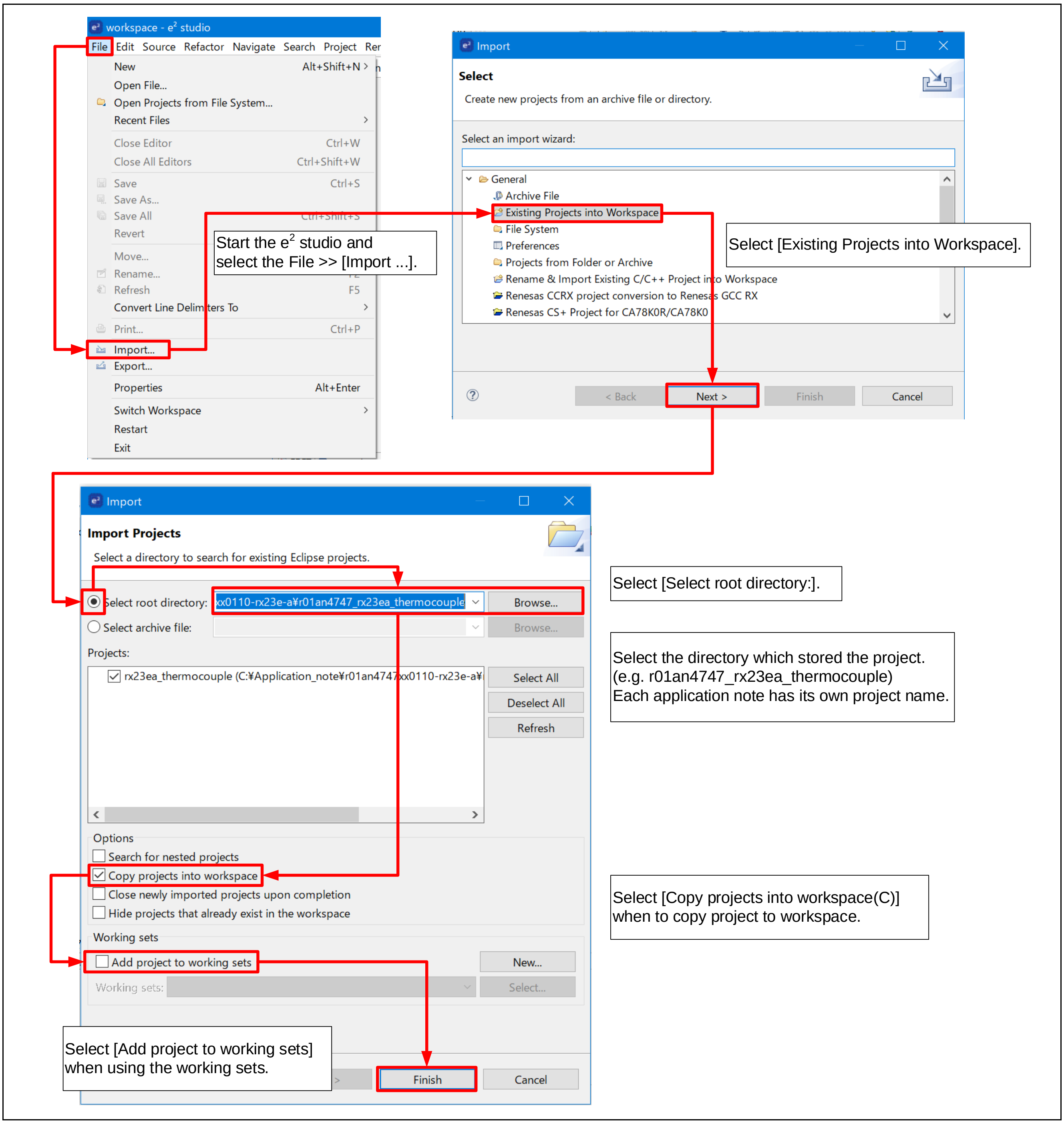


**Figure 4-1 Import a Project into e² Studio**

## 4.2 Procedure in CS+

To use sample programs in CS+, follow the steps below to import them into CS+. In projects managed by CS+, do not use space codes, multibyte characters, and symbols such as "$", "#", "%" in folder names or paths to them.

(Note that depending on the version of CS+ you are using, the interface may appear somewhat different from the screenshots below.)

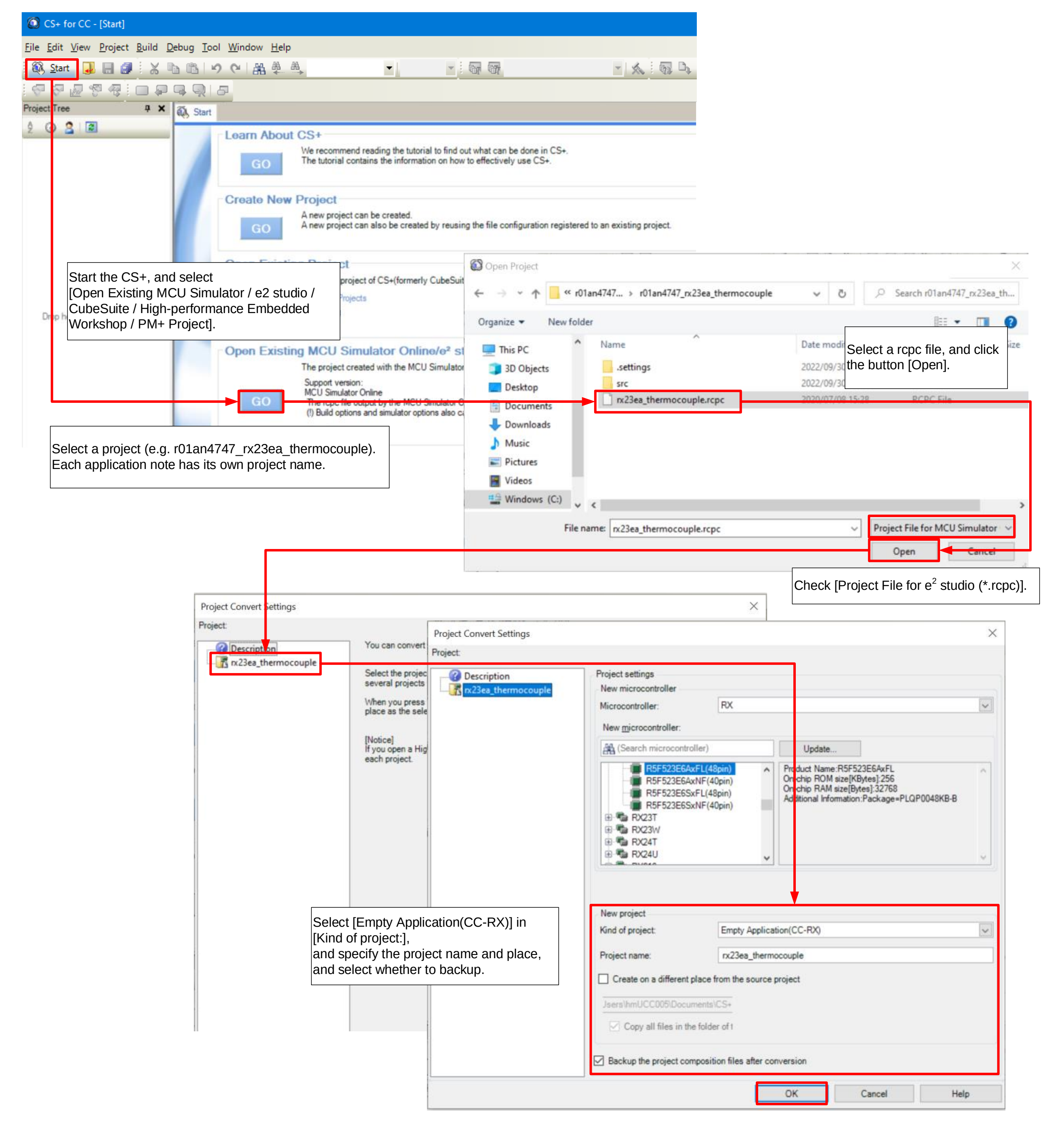


**Figure 4-2 Importing a Project into CS+**

## 5. Start Demonstration

Disconnect the E2 Emulator Lite and turn on the RX671 PoC to start the demonstration program. This demonstration program assumes control of the display and settings of a microwave oven. Set the cooking conditions and recipe selections using the touch buttons and touch slider while checking the LCD.

Hereinafter, touch buttons are described as buttons and touch slider is described as slider.

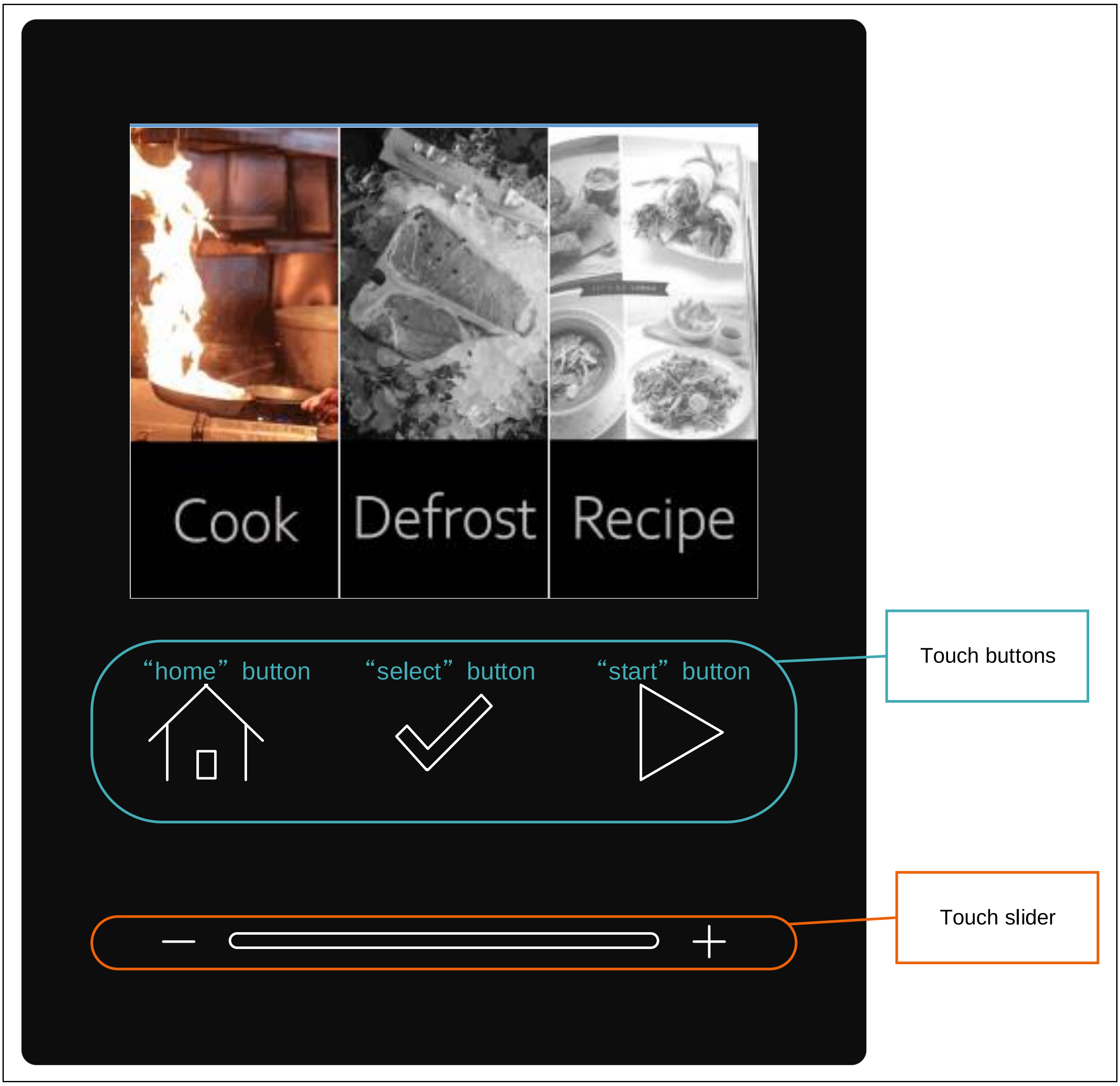


**Figure 5-1 Demonstration screen and operation panel**

## 5.1 Powered on RX671 PoC and menu screen

When RX671 PoC is powered on, the LCD panel displays the RX logo and RX671 features (initial screen) for approximately 5 seconds. When the display finishes, the sample program starts and becomes a menu screen.

And while the initial screen is displayed, can immediately move to the menu screen by touching any button.

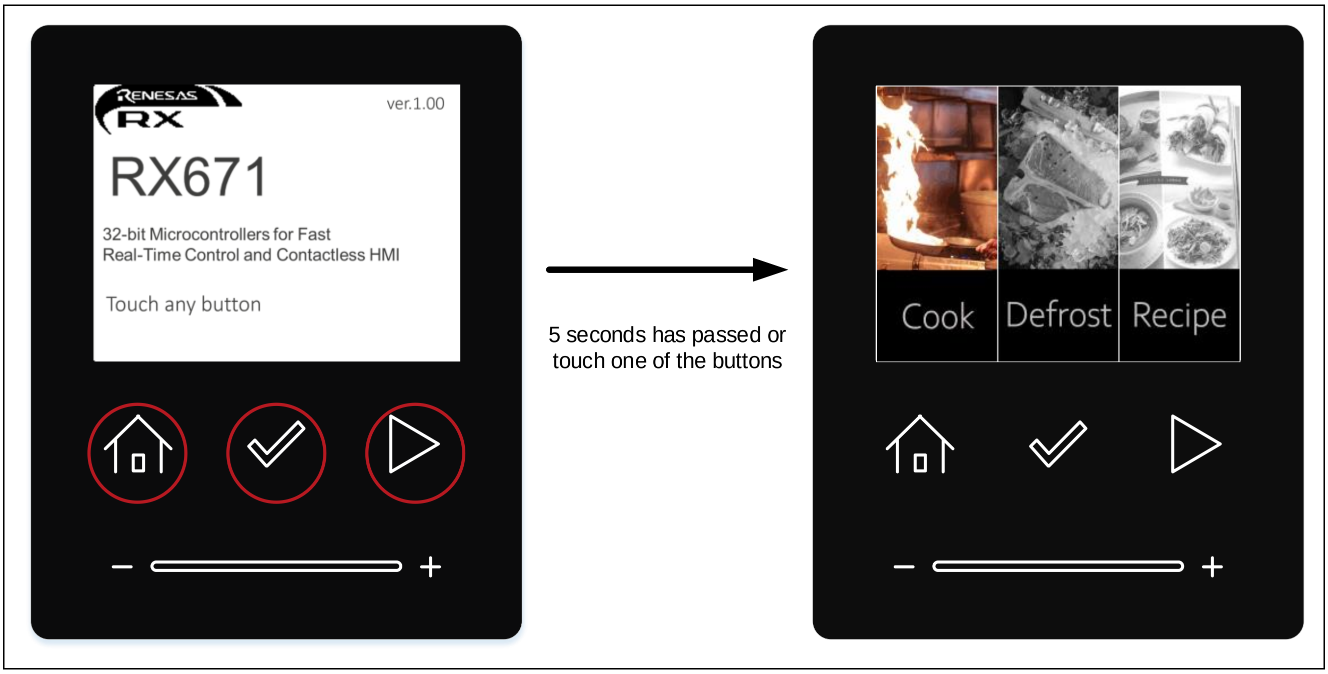


**Figure 5-2 Start of the demonstration**

## 5.2 Menu screen

"Cook", "Defrost" or "Recipe" can be selected with the slider operation on the menu screen.

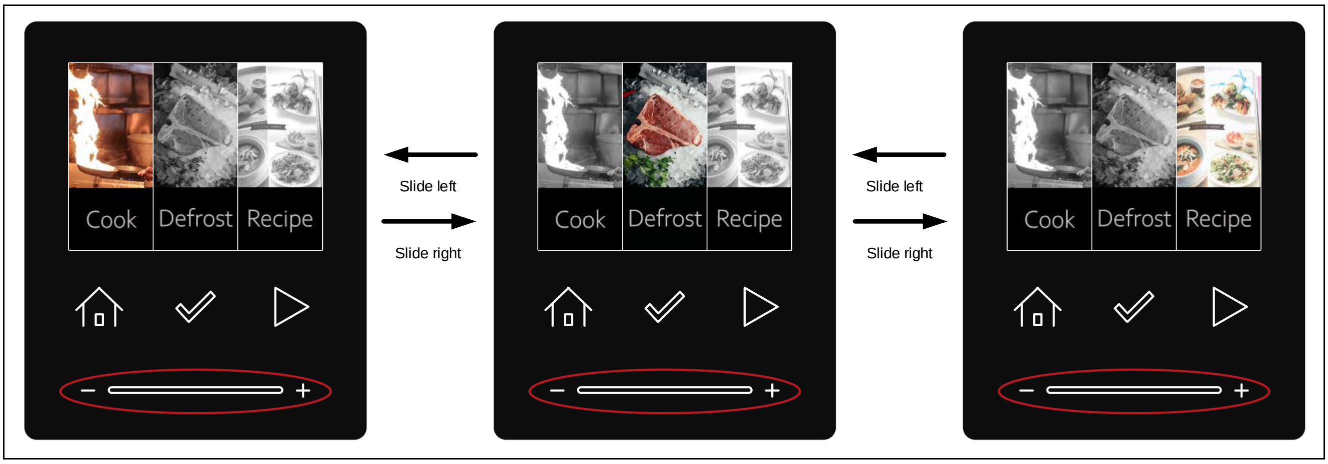


**Figure 5-3 How to operate the menu screen**

## 5.3 Cook setting

### 5.3.1 Move to mode selection screen

While "Cook" is selected on the menu screen, touching the "select" button can move to the Cook mode selection screen.

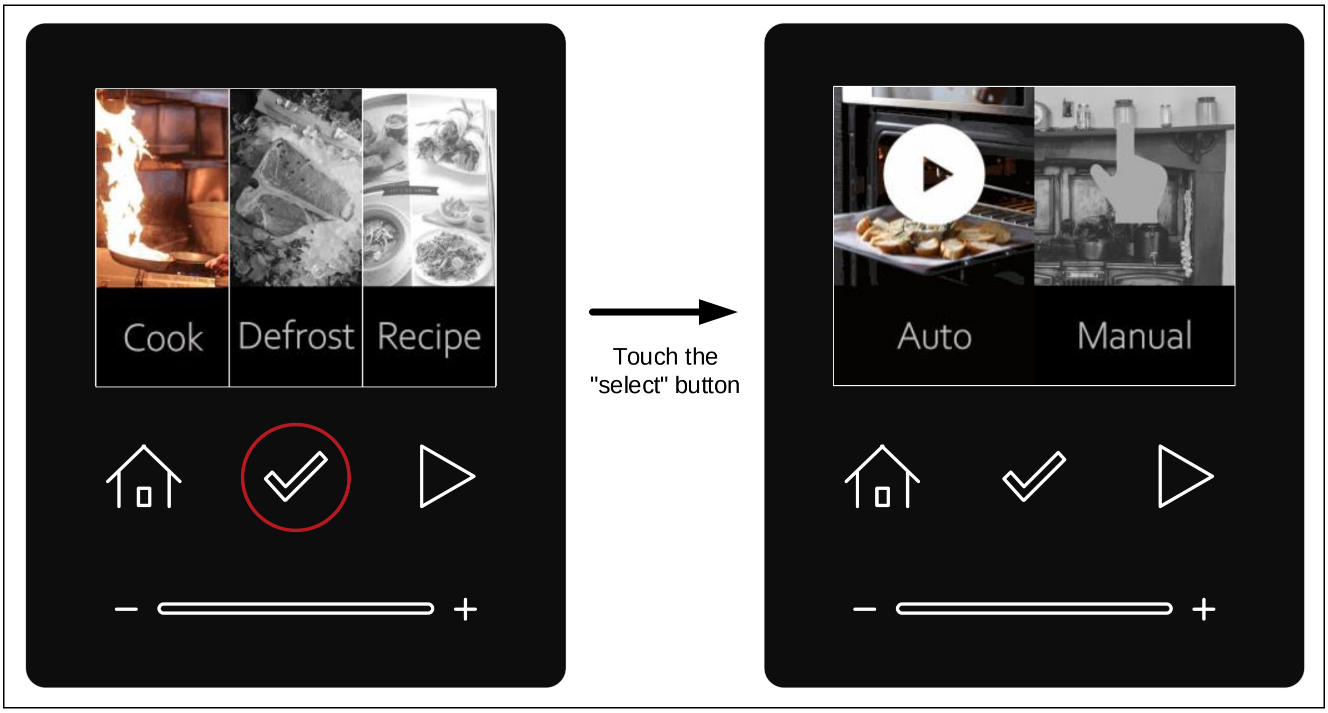


**Figure 5-4 Move to the Cook mode selection screen**

### 5.3.2 Select mode

While the Cook mode selection screen is displayed, "Auto" or "Manual" can be selected with the slider operation.

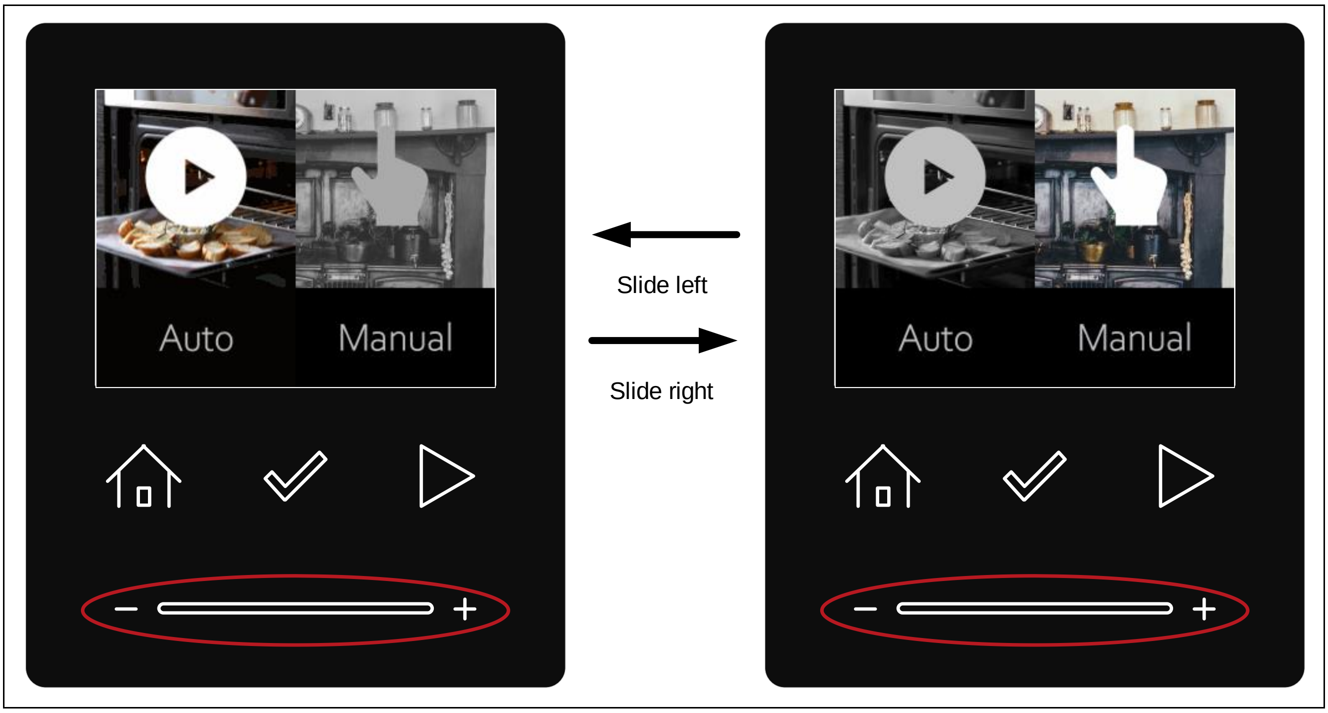


**Figure 5-5 How to operate the Cook mode selection screen**

### 5.3.3 Select Auto

While "Auto" is selected on the Cook mode selection screen, touching the "start" button can start cooking.

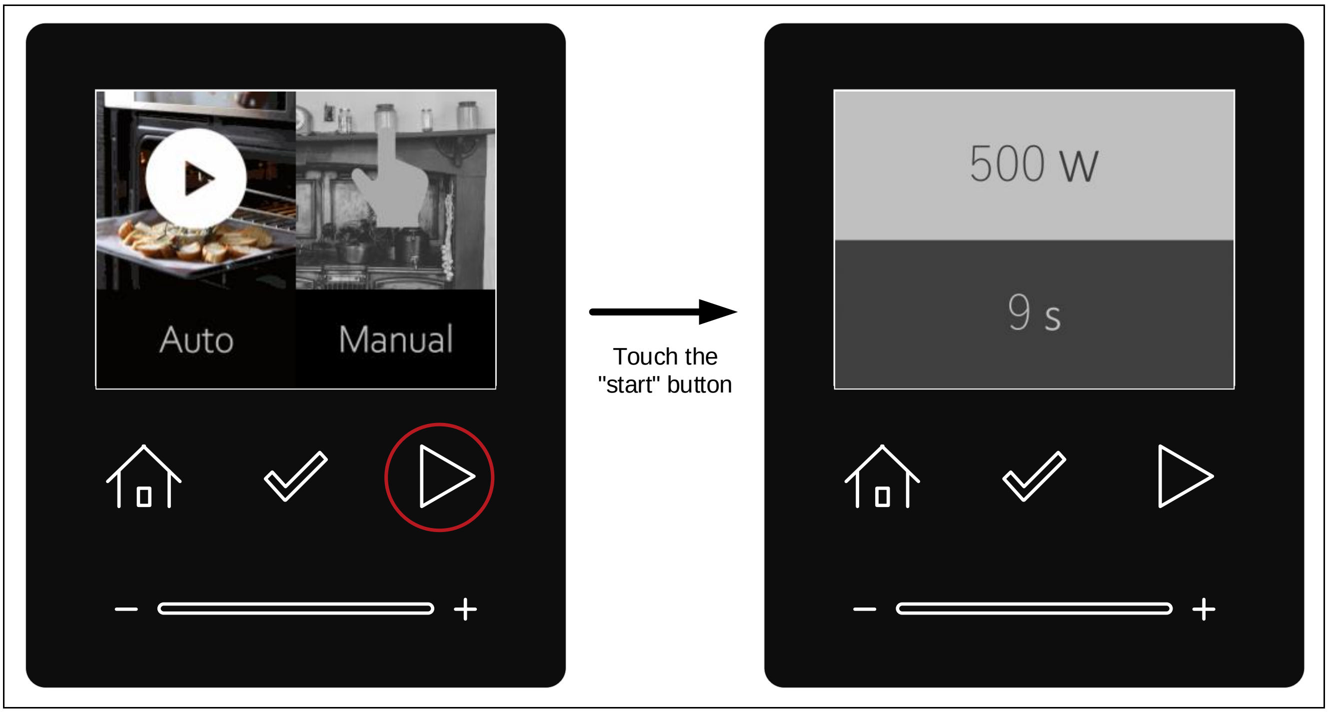


**Figure 5-6 Start cooking in Auto mode**

### 5.3.4 Select Manual

While "Manual" is selected on the Cook mode selection screen, touching the "select" button can move to the Cook detail setting screen.

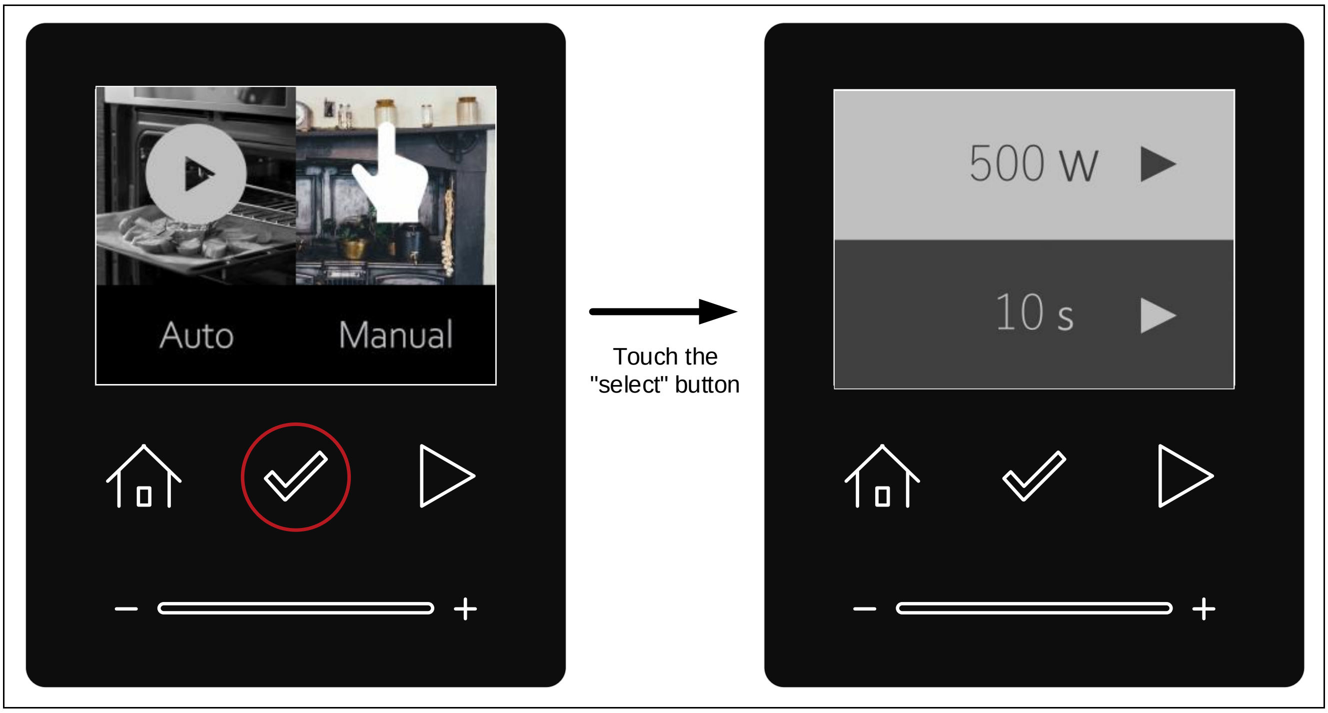


**Figure 5-7 Move to the Cook detail setting screen**

#### 5.3.4.1 Set the number of watts

While the cursor is on the upper side, the number of watts can be set with the slider.
"500W", "600W" and "700W" can be selected as the power level.

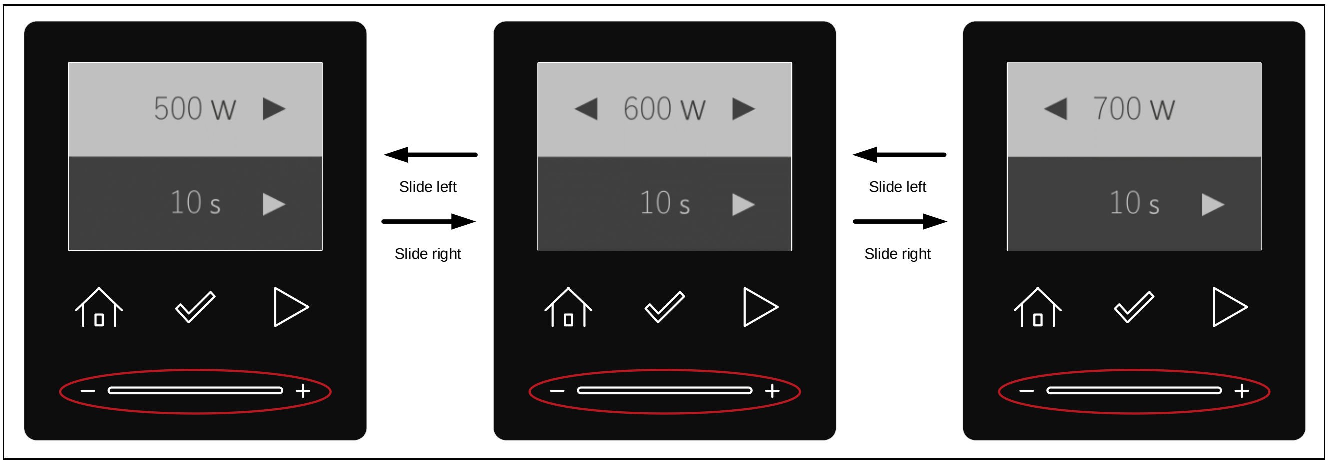


**Figure 5-8 Setting the number of watts**

#### 5.3.4.2 Move the cursor

While the Cook detail setting screen is displayed, touching the "select" button can move the cursor.
The item with a light-colored background is selected.

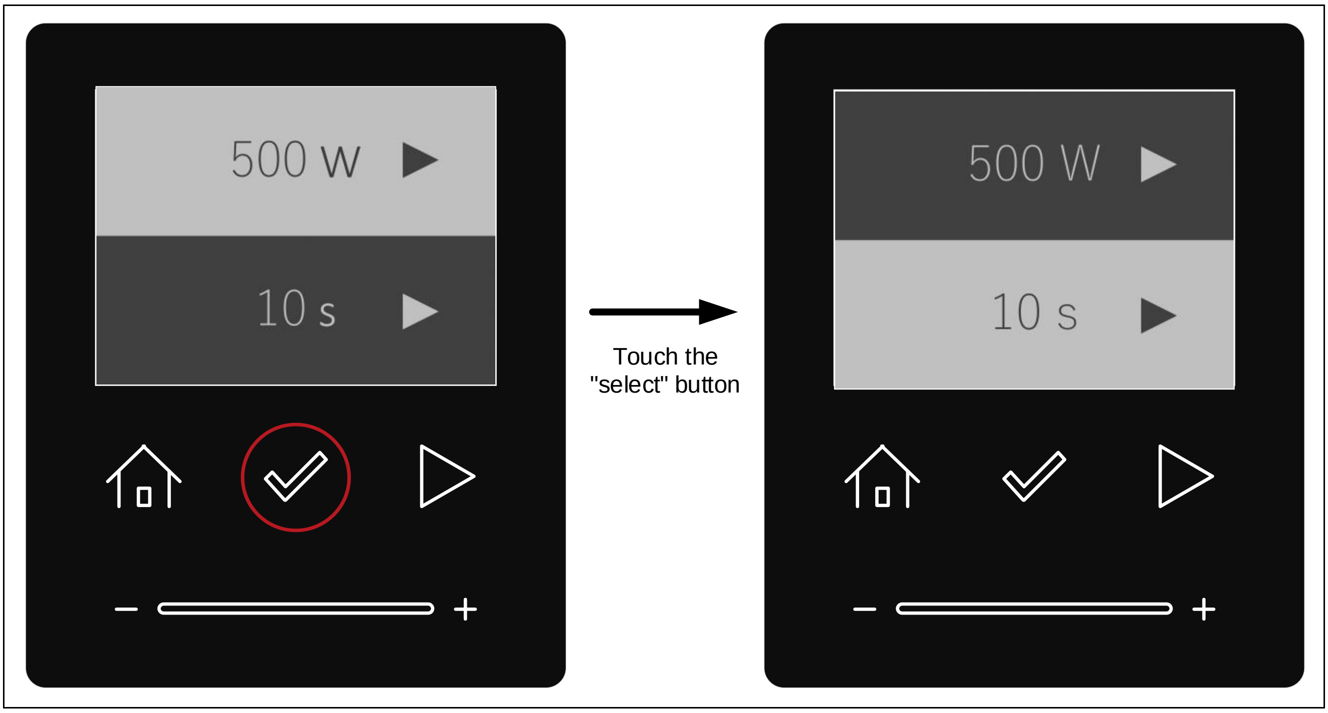


**Figure 5-9 How to operate the cursor on the Cook detail setting screen**

#### 5.3.4.3 Set the number of seconds

While the cursor is on the lower side, the number of seconds can be set with the slider.
"10s", "20s" and "30s" can be selected as the cooking time.

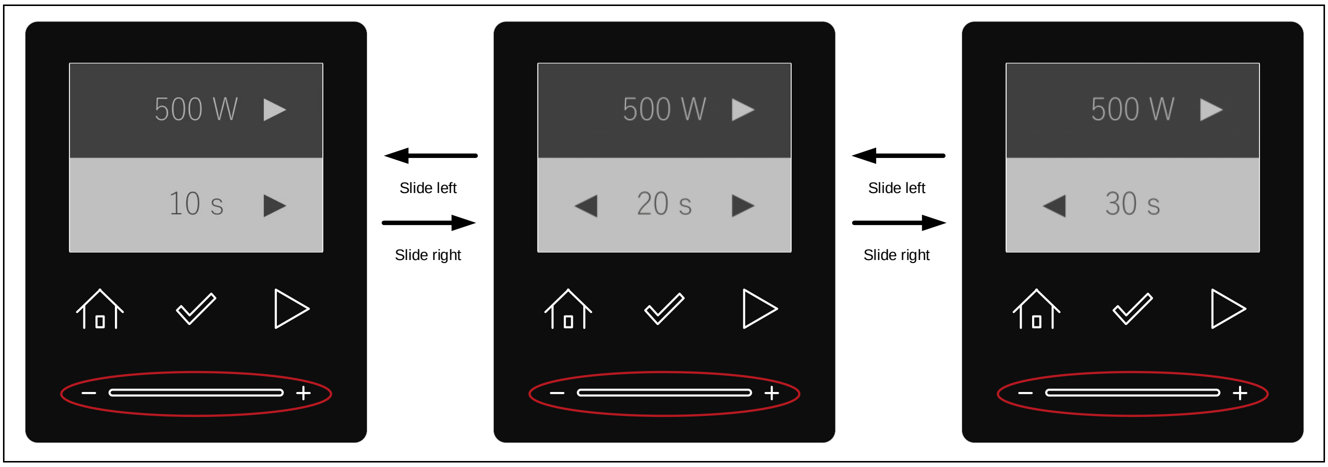


**Figure 5-10 Setting the number of seconds**

#### 5.3.4.4 Start cooking

While the Cook detail setting screen is displayed and the cursor is on the lower side, touching the "start" button can start cooking.

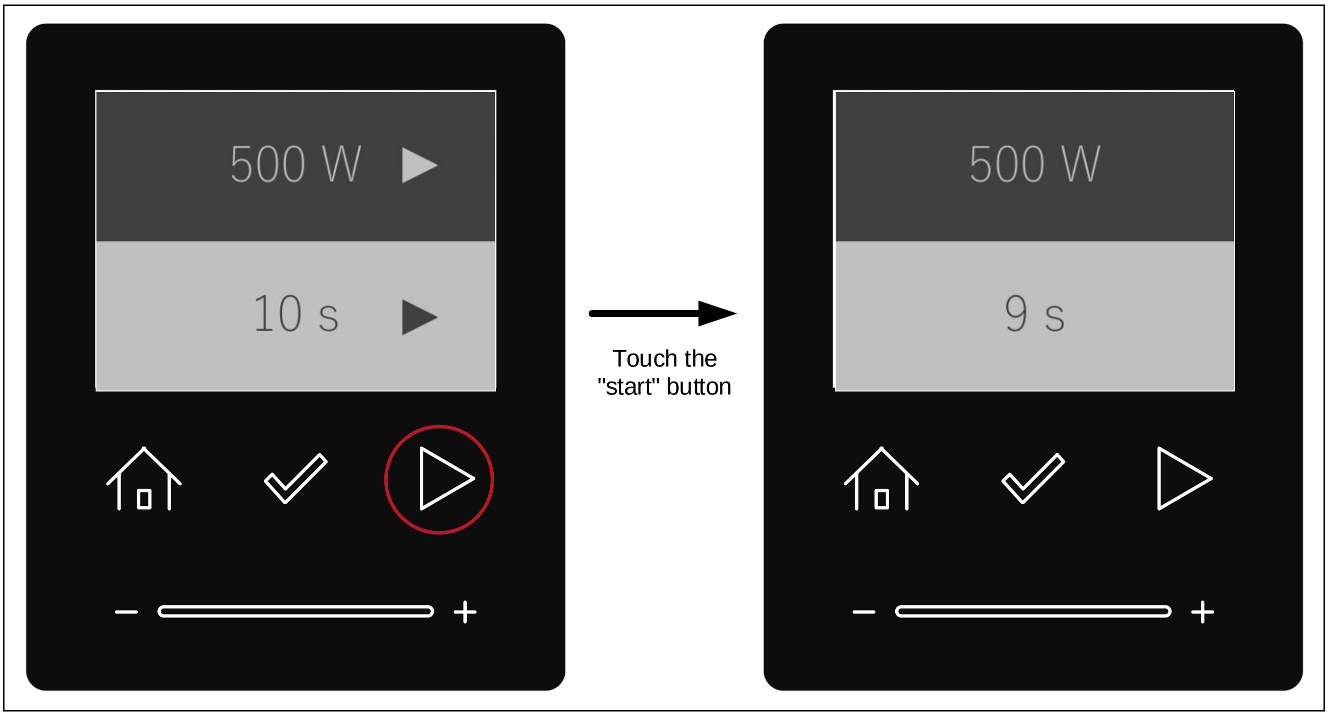


**Figure 5-11 Start cooking in Manual mode**

## 5.4 Defrost setting

### 5.4.1 Move to mode selection screen

While "Defrost" is selected on the menu screen, touching the "select" button can move to the Defrost mode selection screen.

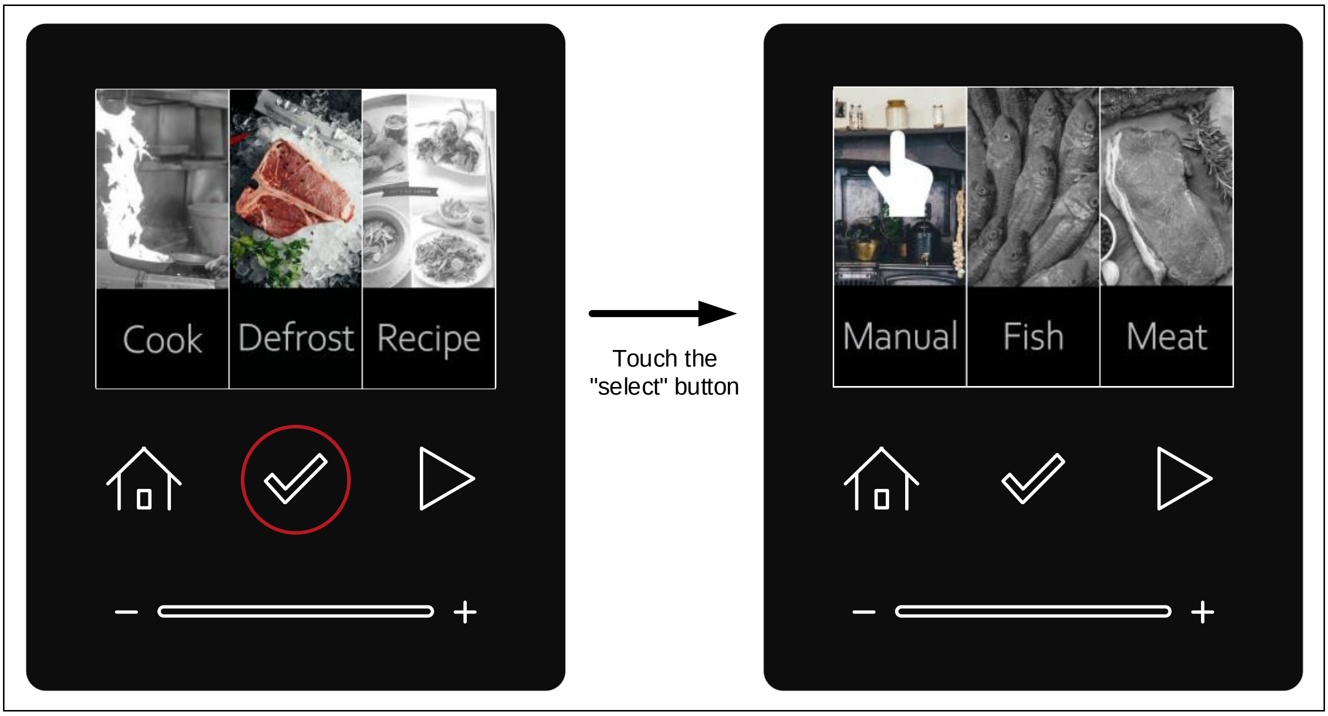


**Figure 5-12 Move to the Defrost mode selection screen**

### 5.4.2 Select mode

While the Defrost mode selection screen is displayed, "Manual", "Fish" or "Meat" can be selected with the slider operation.

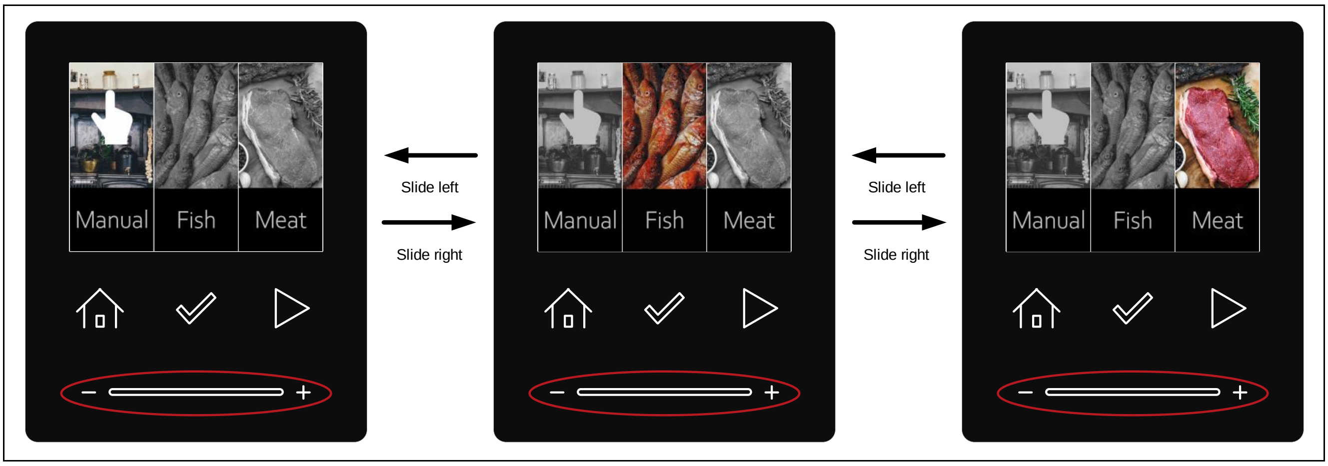


**Figure 5-13 How to operate the Defrost mode selection screen**

### 5.4.3 Select Manual

While "Manual" is selected on the Defrost mode selection screen, touching the "select" button can move to the Defrost detail setting screen.

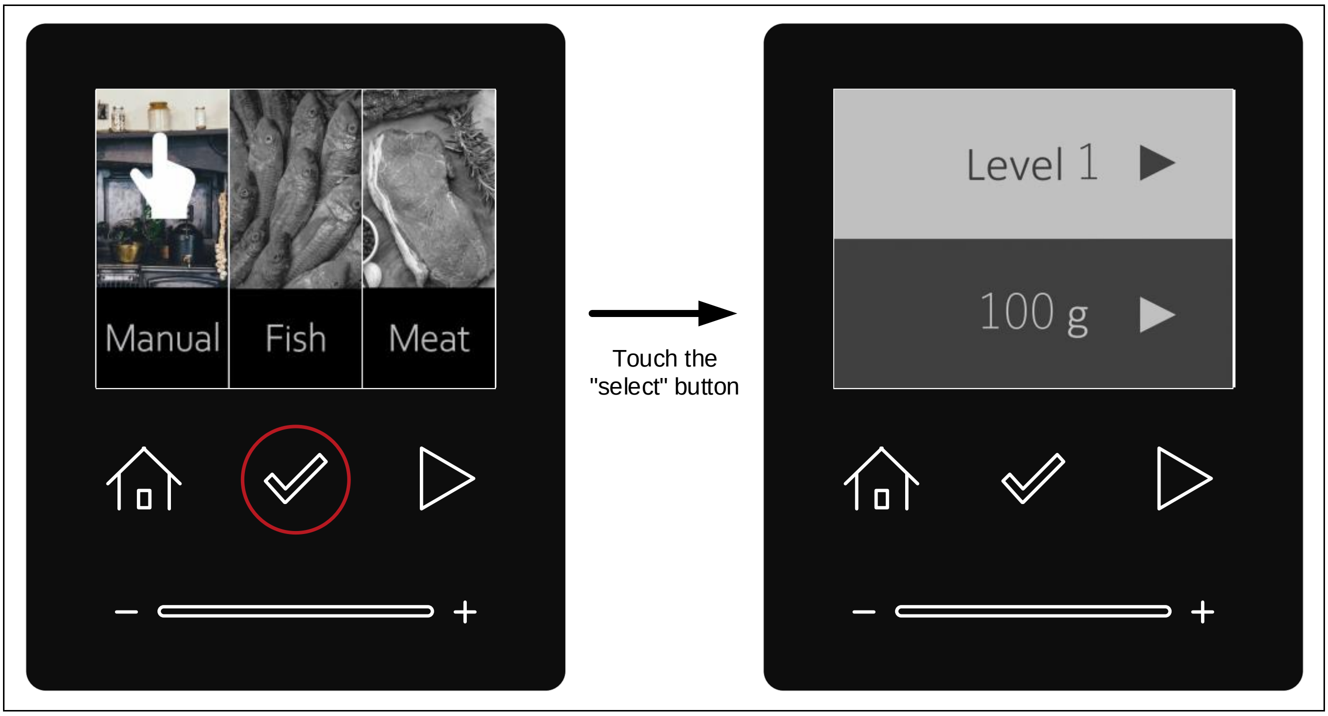


**Figure 5-14 Move to the Defrost detail setting screen**

#### 5.4.3.1 Set the level of defrosting

While the cursor is on the upper side, the level of defrosting can be set with the slider.
"Level1", "Level2" and "Level3" can be selected as the defrosting level.

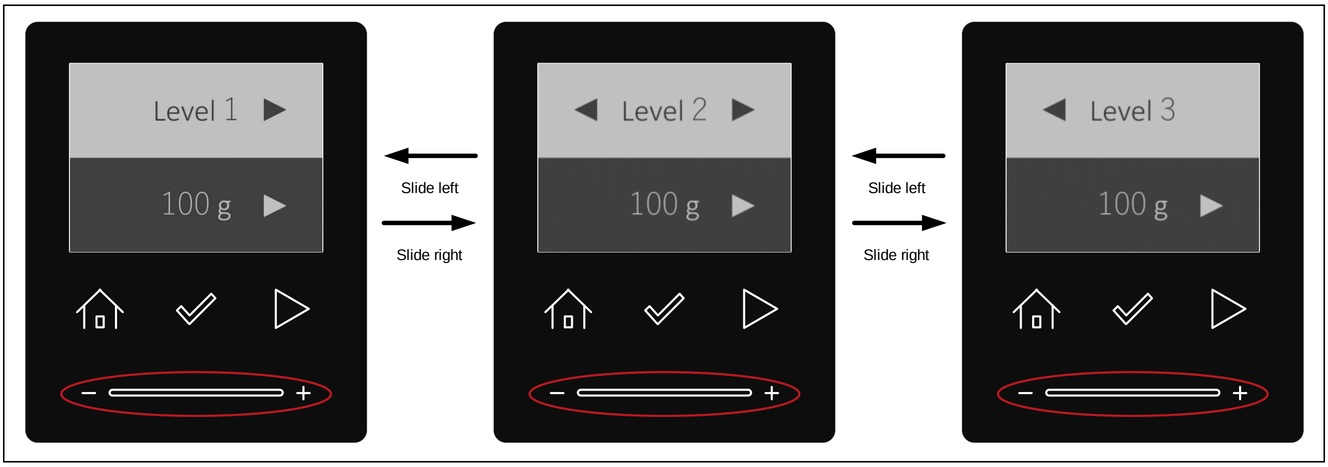


**Figure 5-15 Setting the level of defrosting**

#### 5.4.3.2 Move the cursor

While the Defrost detail setting screen is displayed, touching the "select" button can move the cursor.
The item with a light-colored background is selected.

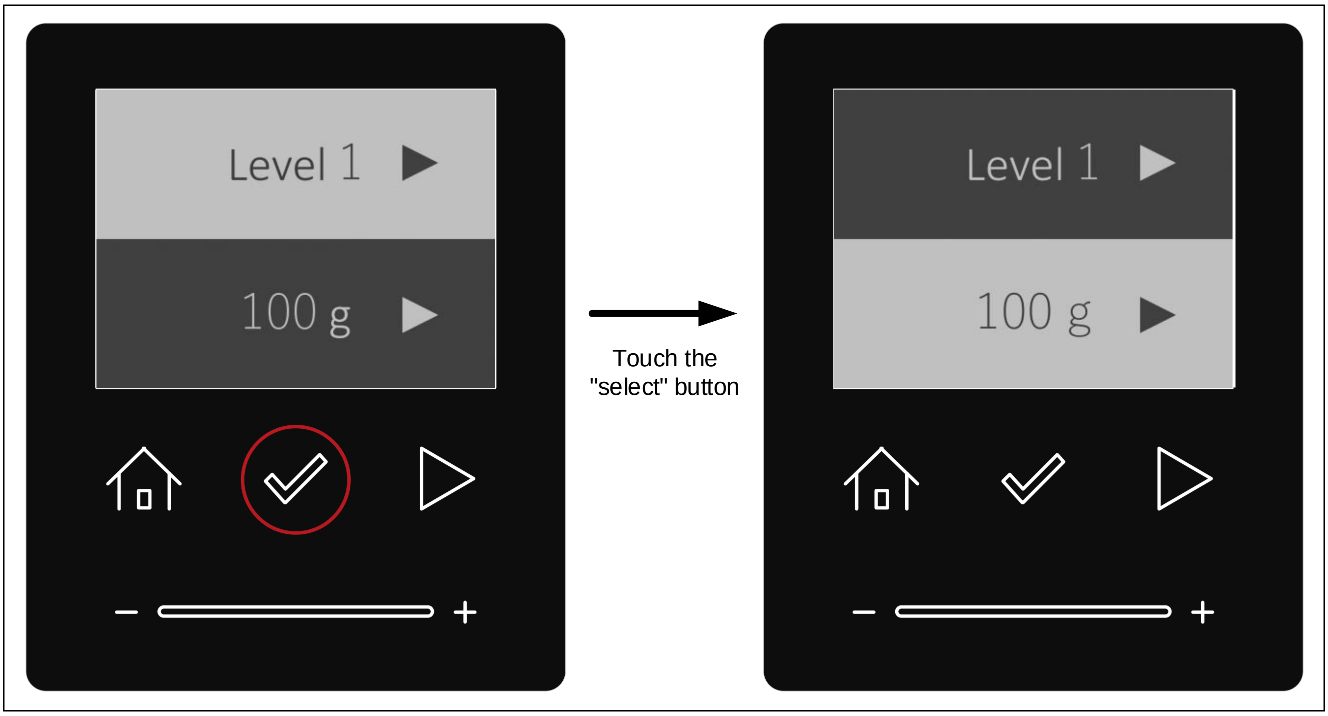


**Figure 5-16 How to operate the cursor on the Defrost detail setting screen**

#### 5.4.3.3 Set the number of grams

While the cursor is on the lower side, the number of grams can be set with the slider.
"100g", "200g" and "300g" can be selected as the defrosting amount.

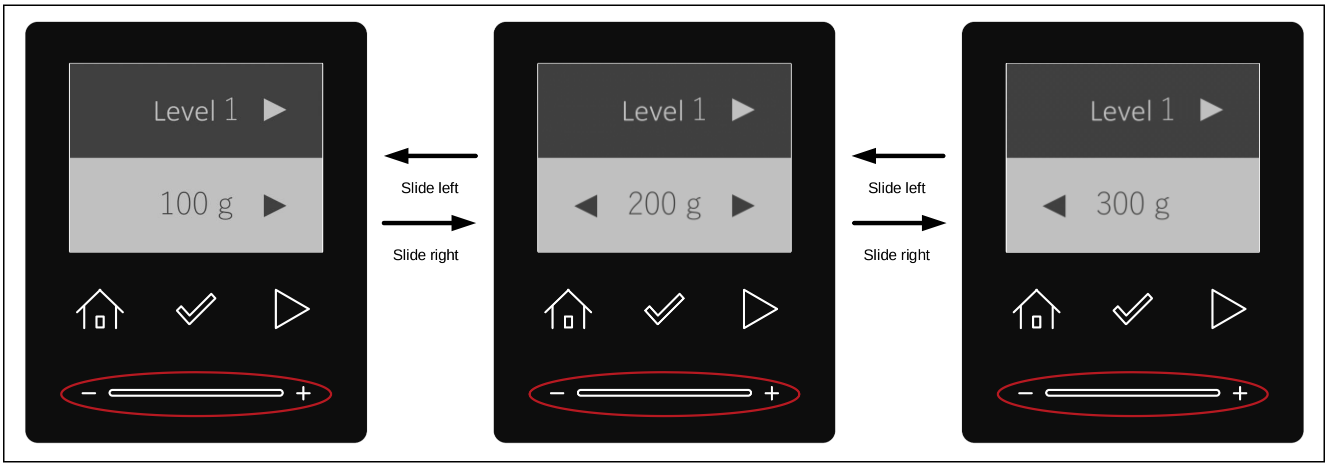


**Figure 5-17 Setting the number of grams**

#### 5.4.3.4 Start defrosting

While the Defrost detail setting screen is displayed and the cursor is on the lower side, touching the "start" button can start defrosting.

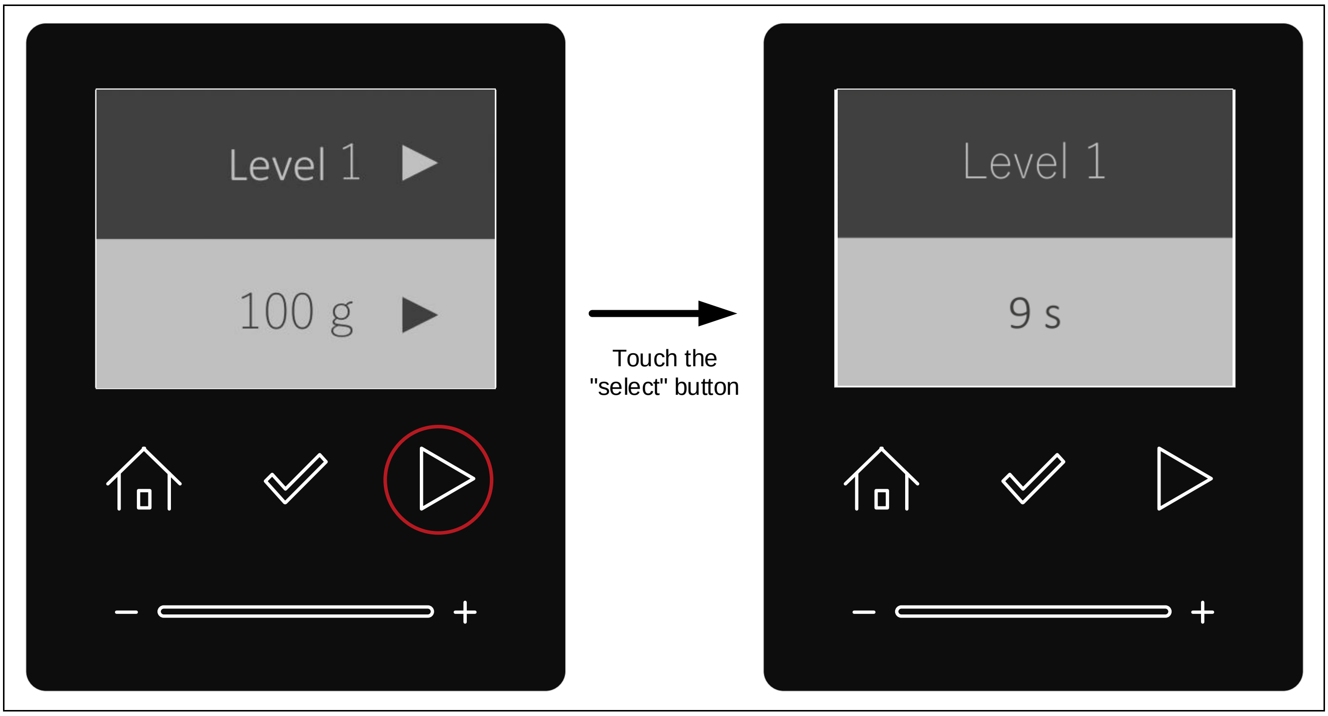


**Figure 5-18 Start defrosting in Manual mode**

### 5.4.4 Select Fish

While "Fish" is selected on the Defrost mode selection screen, touching the "start" button can start defrosting with the settings for "Fish".

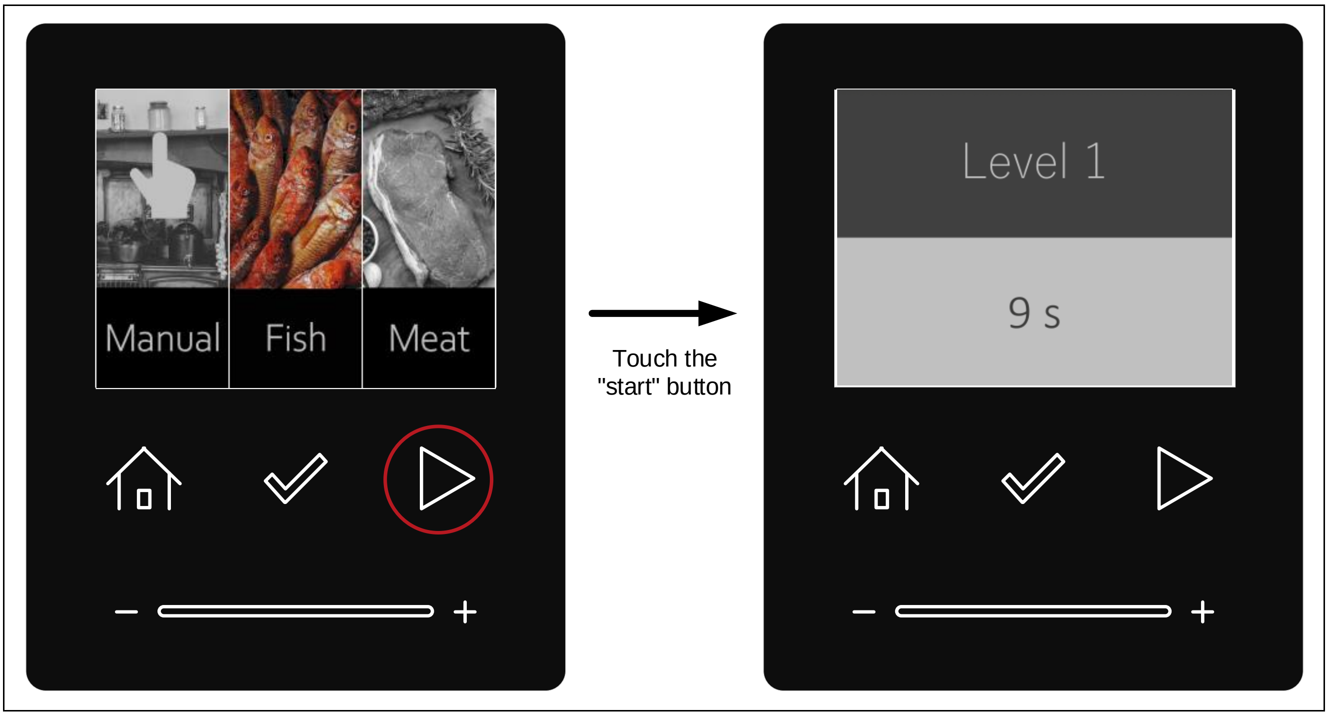


**Figure 5-19 Start defrosting in Fish mode**

### 5.4.5 Select Meat

While "Meat" is selected on the Defrost mode selection screen, touching the "start" button can start defrosting with the settings for "Meat".

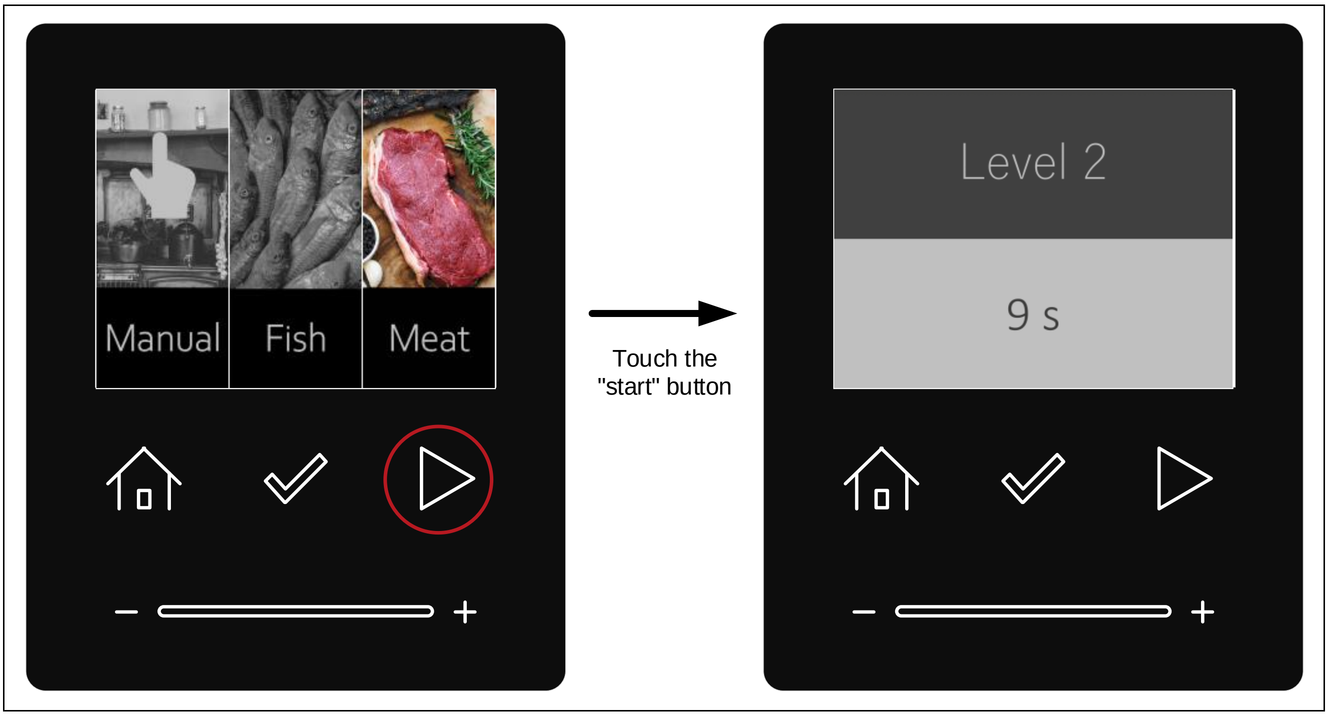


**Figure 5-20 Start defrosting in Meat mode**

## 5.5 Recipe setting

Recipe mode is not available with the firmware ver.0.90.

### 5.5.1 Move to recipe selection screen

While "Recipe" is selected on the menu screen, touching the "select" button can move to the Recipe selection screen.

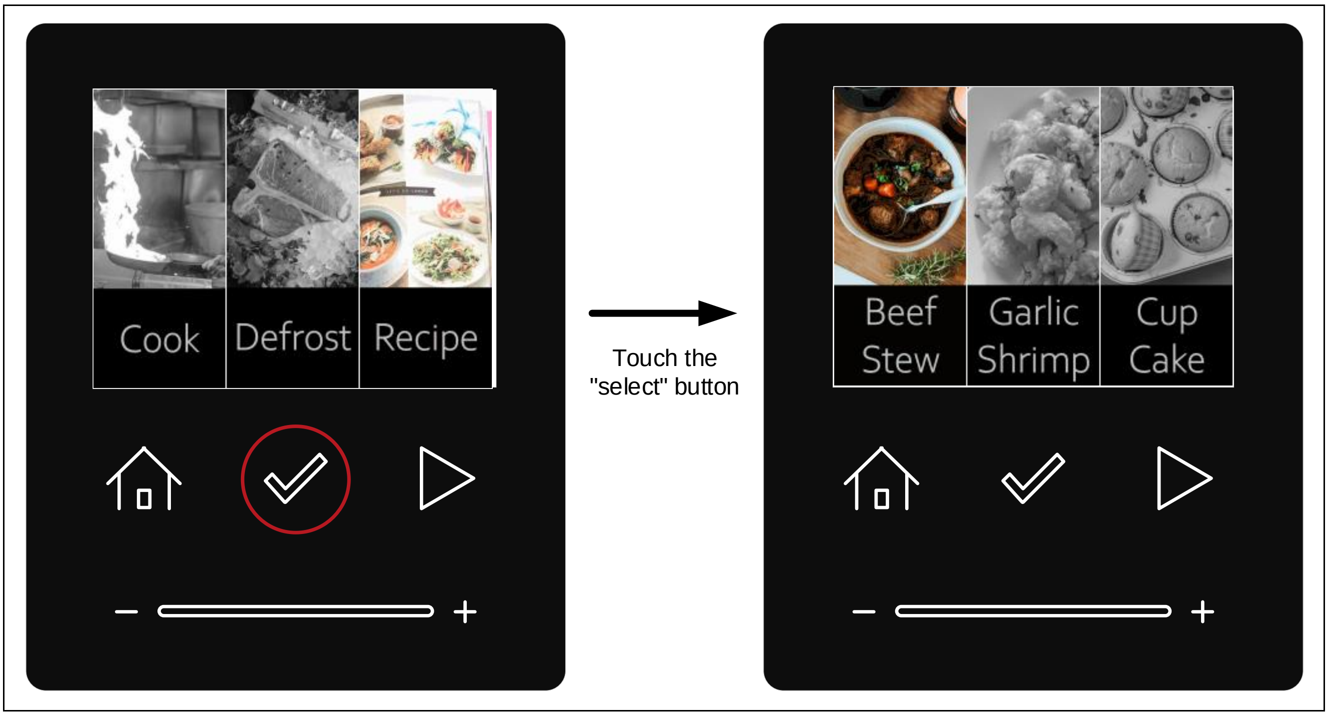


**Figure 5-21 Move to the Recipe selection screen**

### 5.5.2 Select recipe

While the Recipe selection screen is displayed, "Beef Stew", "Garlic Shrimp" or "Cup Cake" can be selected with the slider operation.

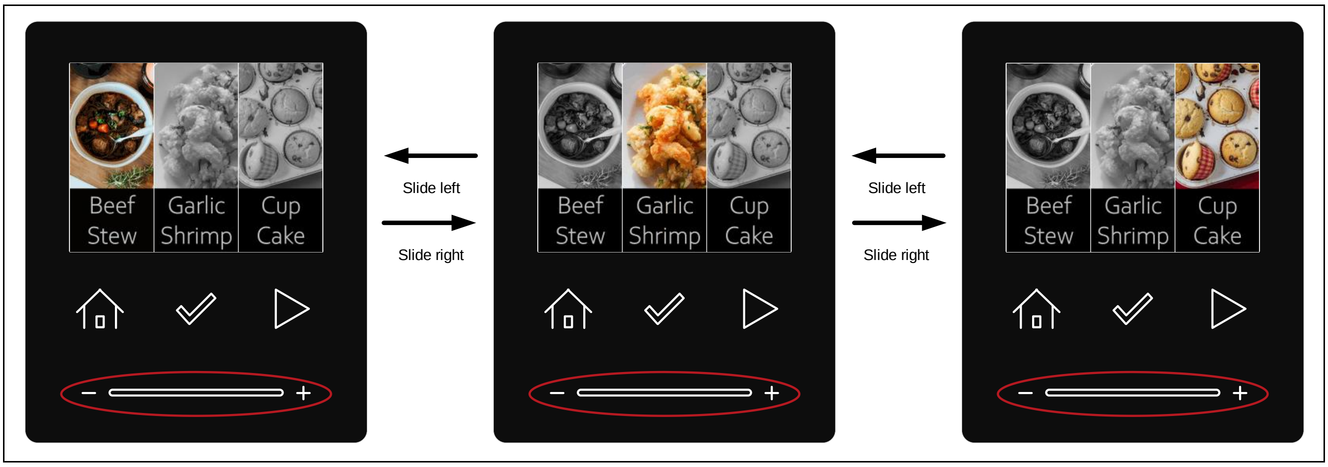


**Figure 5-22 How to operate the Recipe selection screen**

### 5.5.3 Select Beef Stew

While "Beef Stew" is selected on the Recipe selection screen, touching the "start" button can start cooking for the Settings for Beef Stew.

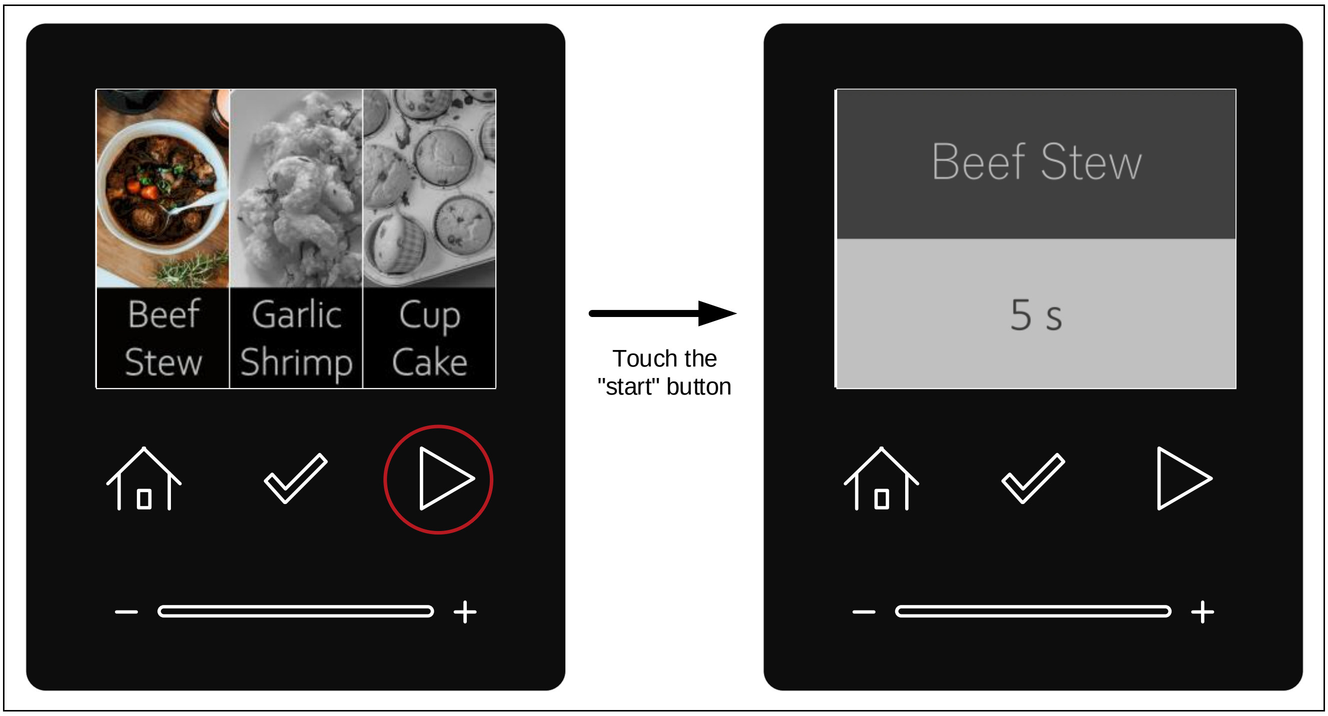


**Figure 5-23 Start cooking in Beef Stew mode**

### 5.5.4 Select Garlic Shrimp

While "Garlic Shrimp" is selected on the Recipe selection screen, touching the "start" button can start cooking for the Settings for Garlic Shrimp.

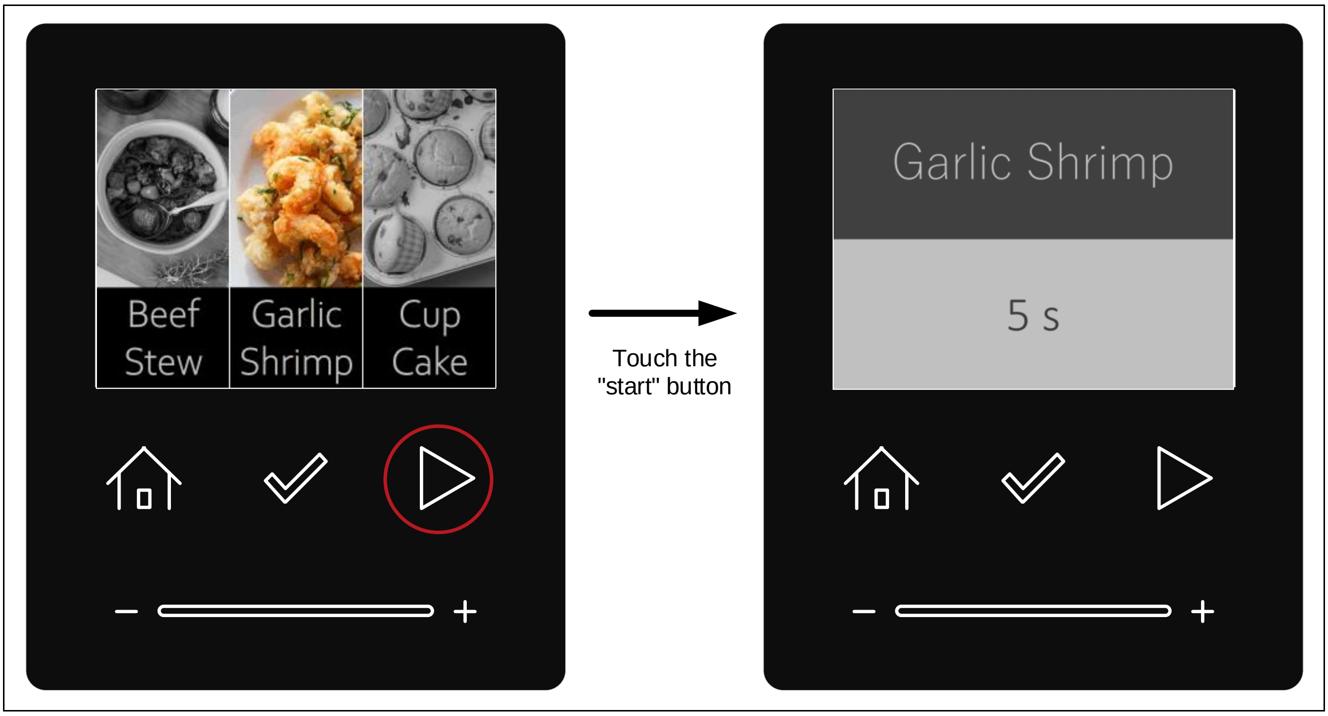


**Figure 5-24 Start cooking in Garlic Shrimp mode**

### 5.5.5 Select Cup Cake

While "Cup Cake" is selected on the Recipe selection screen, touching the "start" button can start cooking for the Settings for Cup Cake.

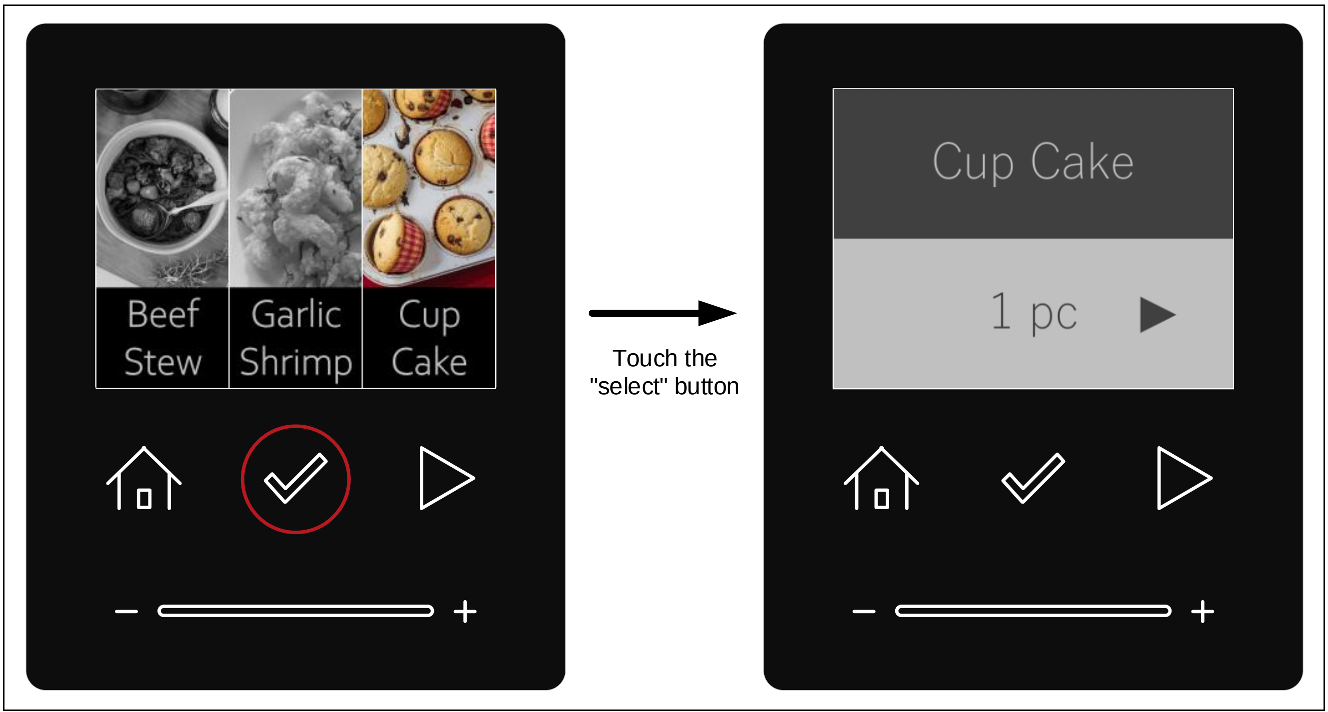


**Figure 5-25 Move to the Cup Cake detail setting screen**

#### 5.5.5.1 Set the number of cupcakes

You can set the number of cupcakes with the slider.
"1pc", "2pcs" and "3pcs" can be selected as the cooking amount.

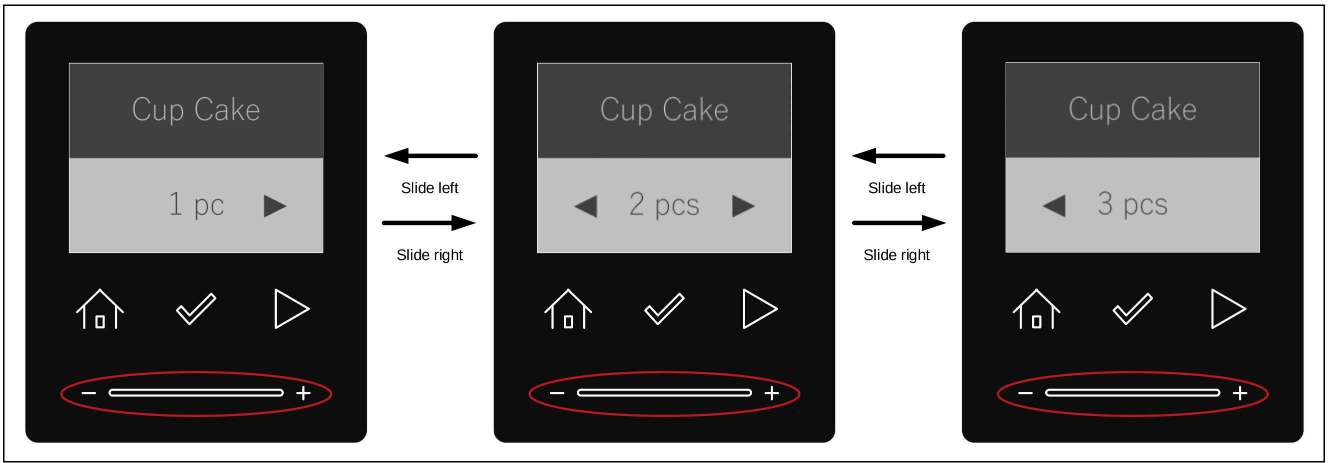


**Figure 5-26 Setting the number of cupcakes**

#### 5.5.5.2 Start cooking

While the Cup Cake detail setting screen is displayed, touching the "start" button can start cooking.

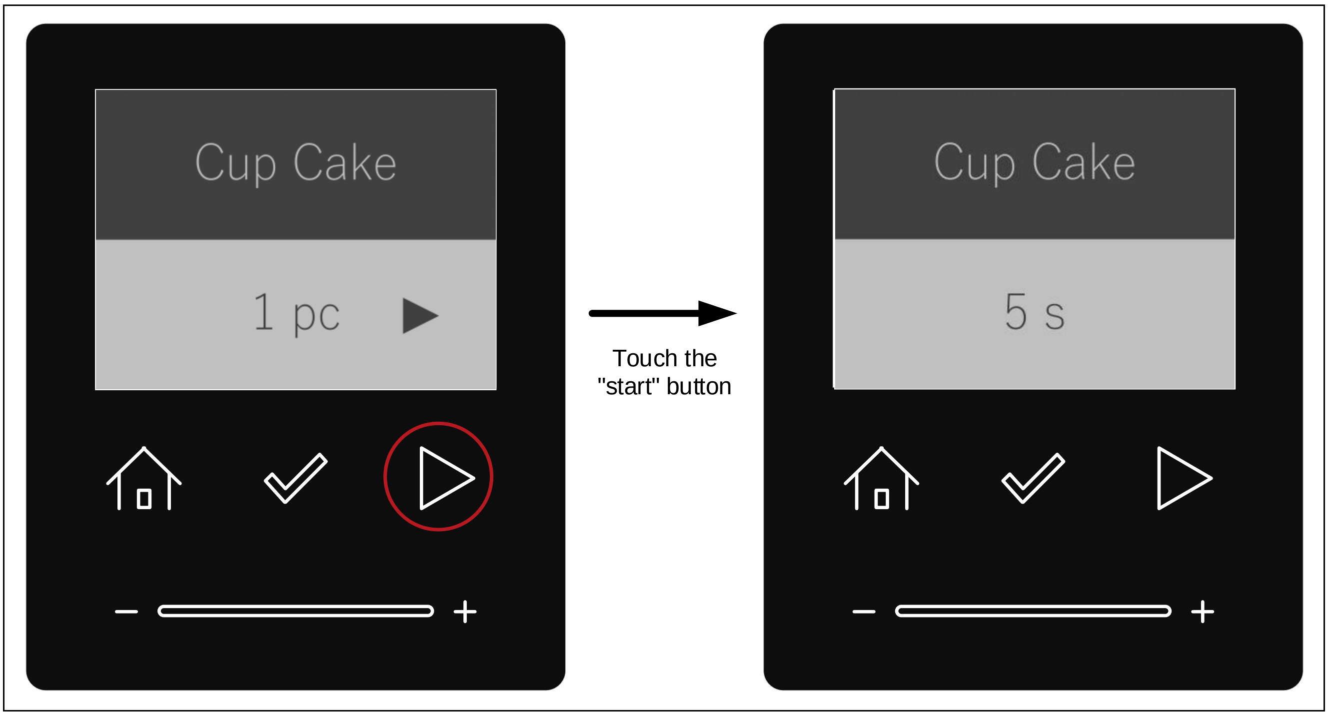


**Figure 5-27 Start cooking in Cup Cake mode**

## 5.6 About the “home” button

The "home" button returns to the menu screen from any screen.

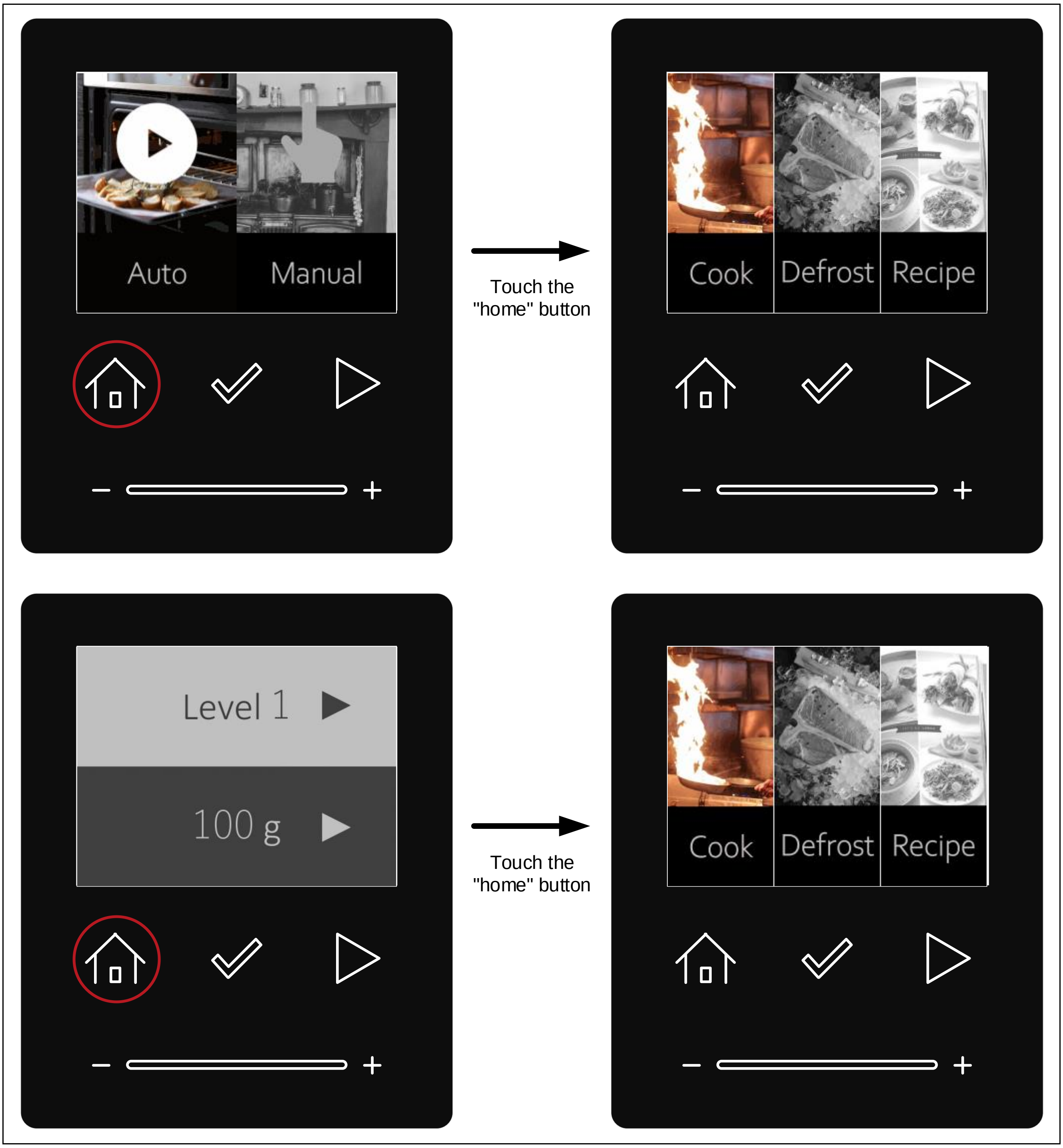


**Figure 5-28 Example of "home" button operation**

## 5.7 About the cooking completion screen

While completed cooking, the cooking completion screen is displayed for 3 seconds. After that, move to the menu screen automatically.

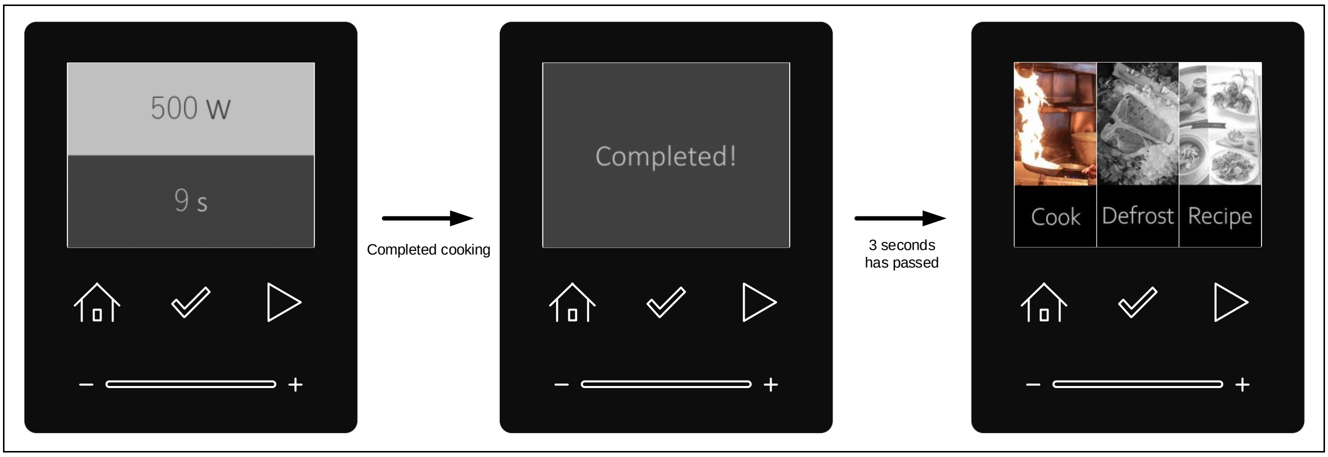


**Figure 5-29 Example of cooking completion operation**

## 5.8 Automatic LCD off function

If no touch operation is performed for 10 seconds, the LCD will turn off. Touching any buttons will return to the previous screen.

## 6. Update firmware version

Two version of firmware are available in this application note. One of them has firmware version 0.90 and its file name is userprog_v0.90.rsu, the other firmware version is 1.00 and its file name is userprog_v1.00.rsu. In the initial state, version 0.90 firmware including a bootloader that supports FOTA is written, and can update it to version 1.00 firmware by following the procedure described later.

The differences by firmware version are shown below. The firmware version in use is displayed on the initial screen. In addition, the menu screen is different for each version.

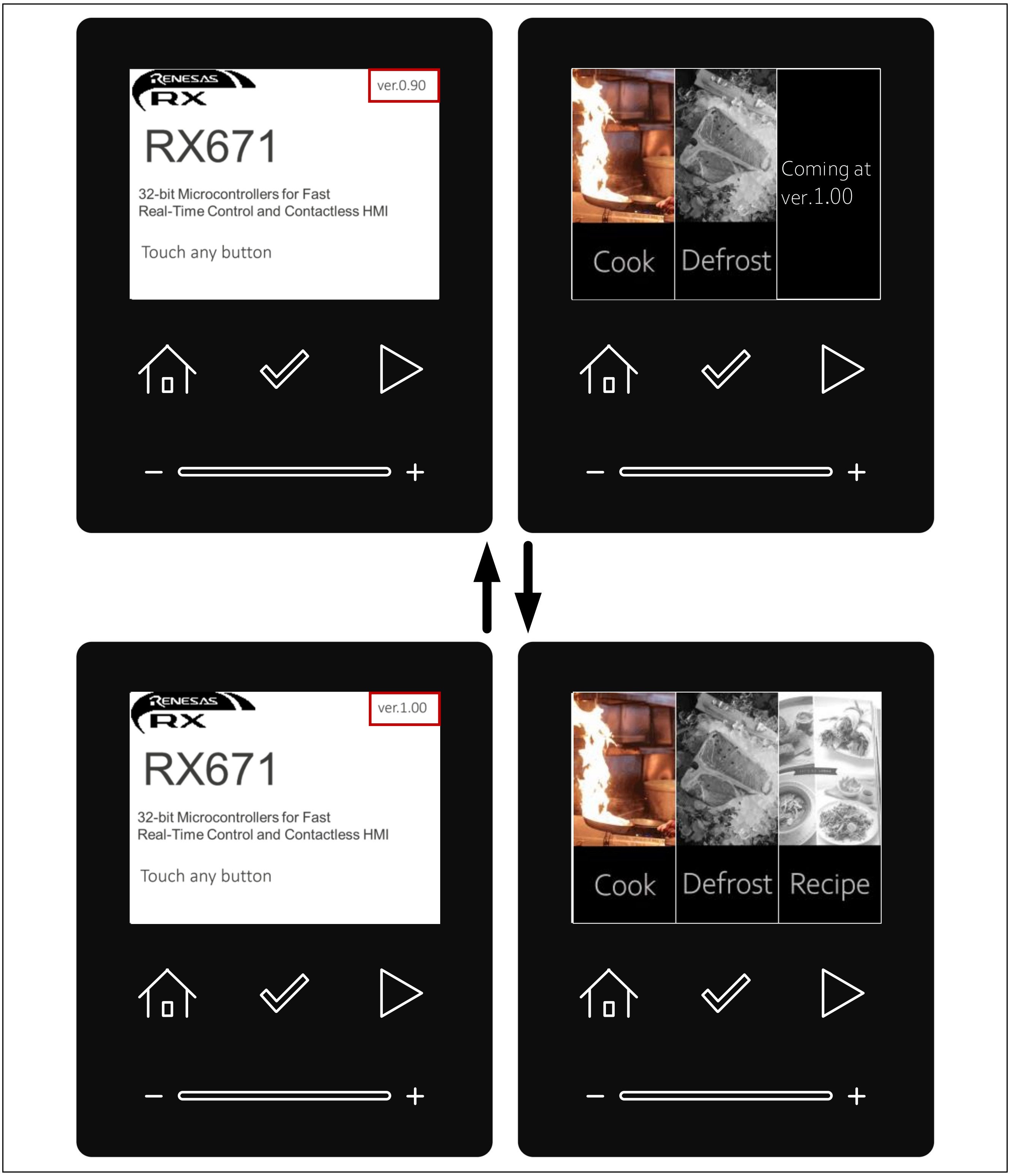


**Figure 6-1 Differences by firmware version**

1. Upload and save the OTA update firmware userprog_v0.90.rsu oruserprog_v1.00.rsu to Amazon S3 bucket as described in "1.2 Create an Amazon S3 bucket" in “How to implement FreeRTOS OTA by using Amazon Web Services on RX65N”.

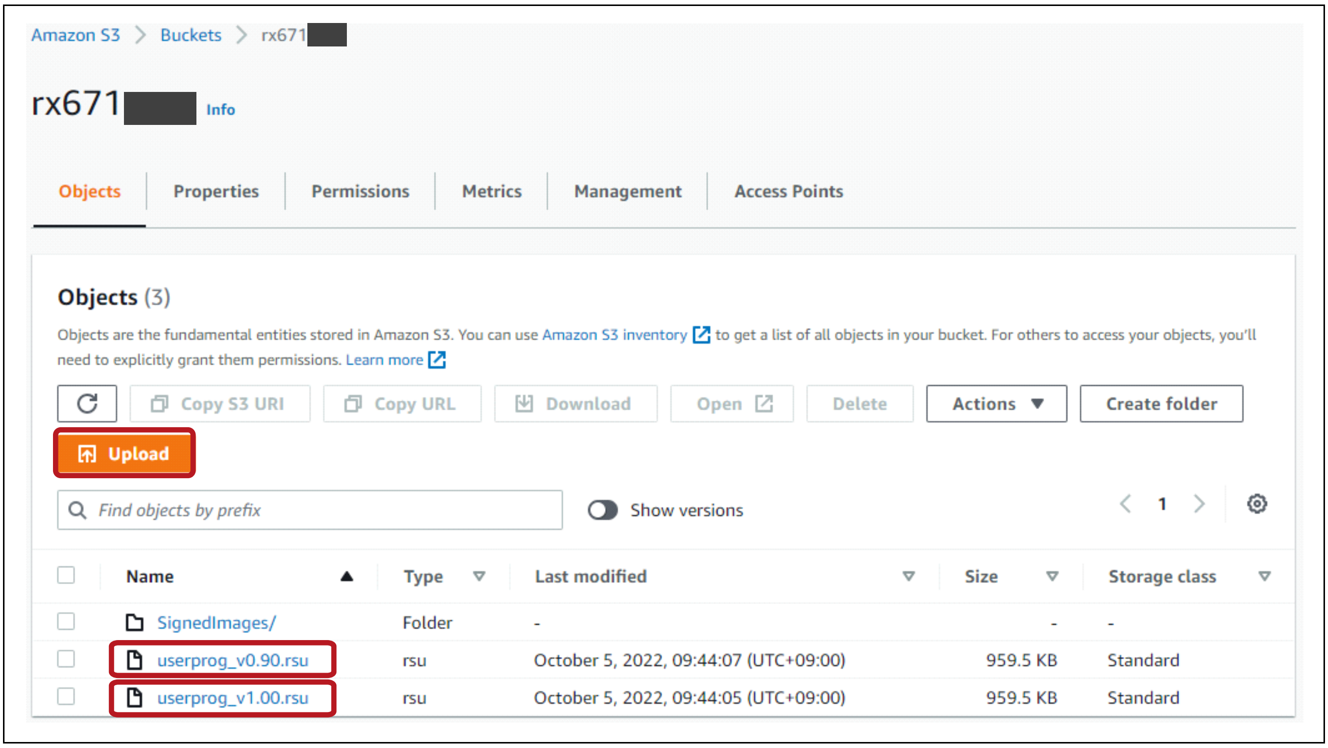


**Figure 6-2 userprog.rsu upload**

2. Create job to update firmware on RX671 PoC.

AWS IoT Jobs is a service that notifies one or more connected devices that they have a pending "job". A job can be used to manage large numbers of devices, update firmware and security certificates on devices, or perform administrative tasks such as rebooting and diagnostics devices.

—Select [AWS IoT] → [Manage] → [Jobs] → [Create] → [Create OTA Update job] → Set job name → [Next]

—Create a FreeRTOS OTA update job as below:

Select the name of the thing. (Figure 6-3 (a), Figure 7-11)
Select Code signing profile. (Figure 6-3 (b))
Select firmware image from S3 for FOTA. (Figure 6-3 (c))
Select IAM role. (Figure 6-3 (d))

—Click [Next]

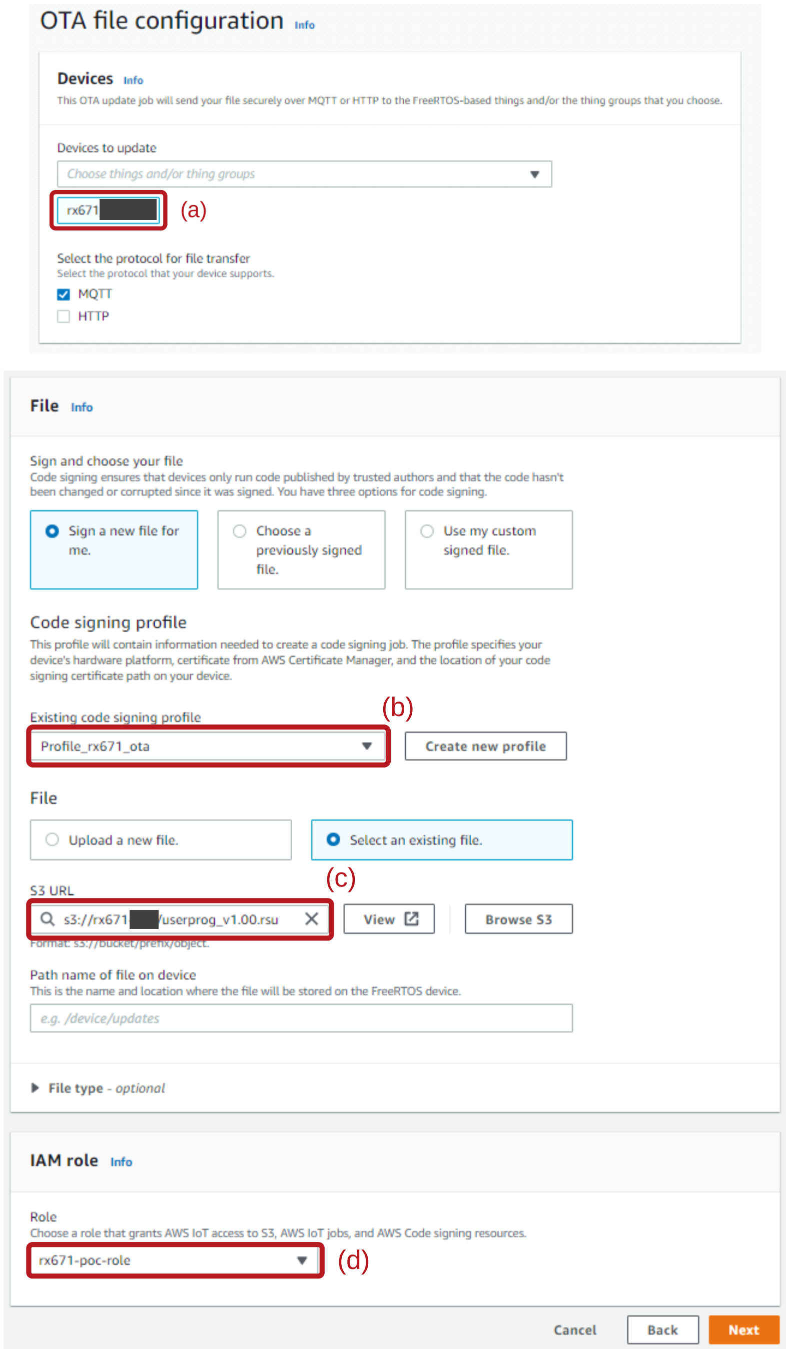


**Figure 6-3 Job creation (1)**

3. Click [Create job].

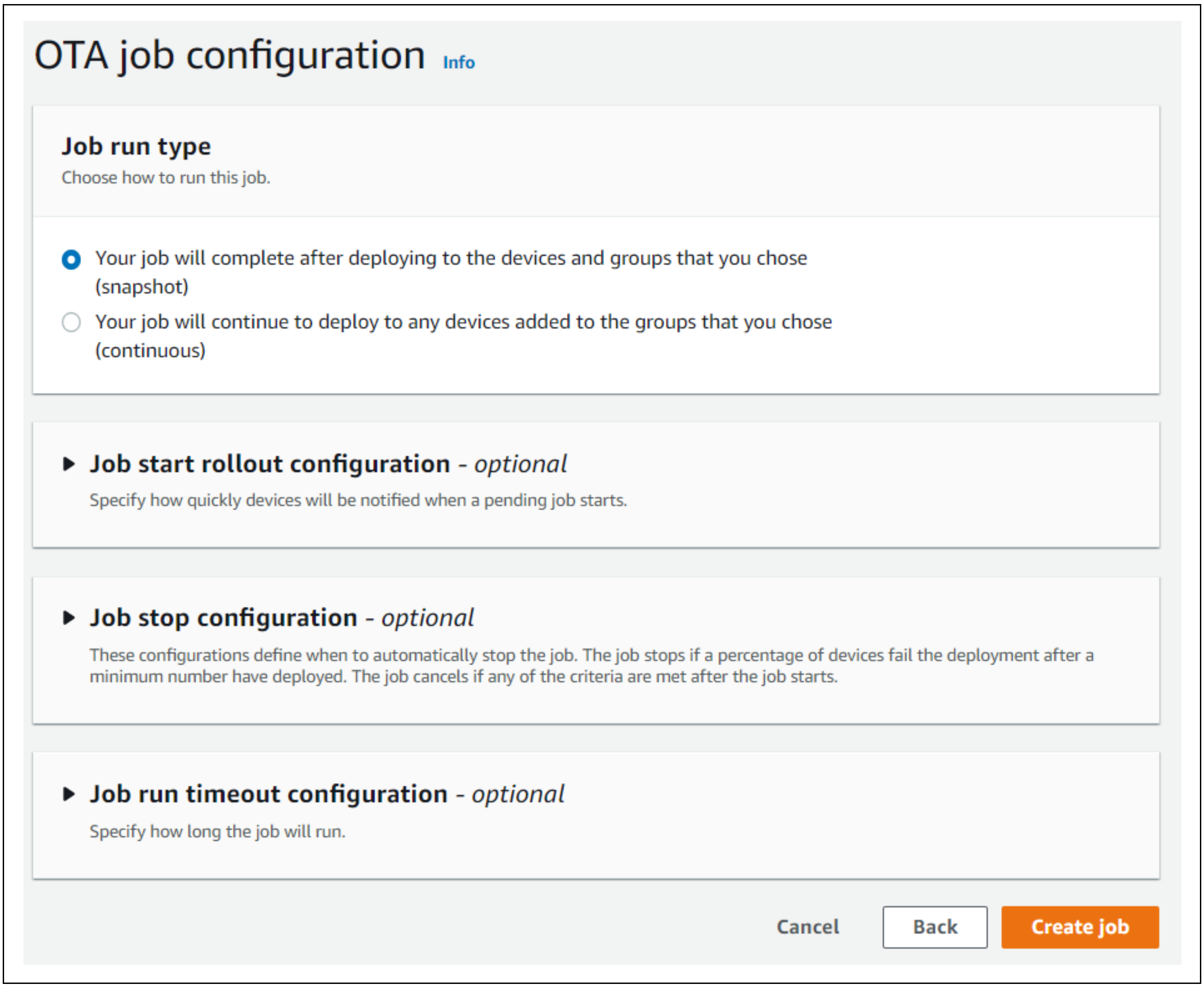


**Figure 6-4 Job creation (2)**

4. Open Tera Term and confirm that the firmware has been updated.
   OTA demonstration version is 1.00 and has been updated successfully.

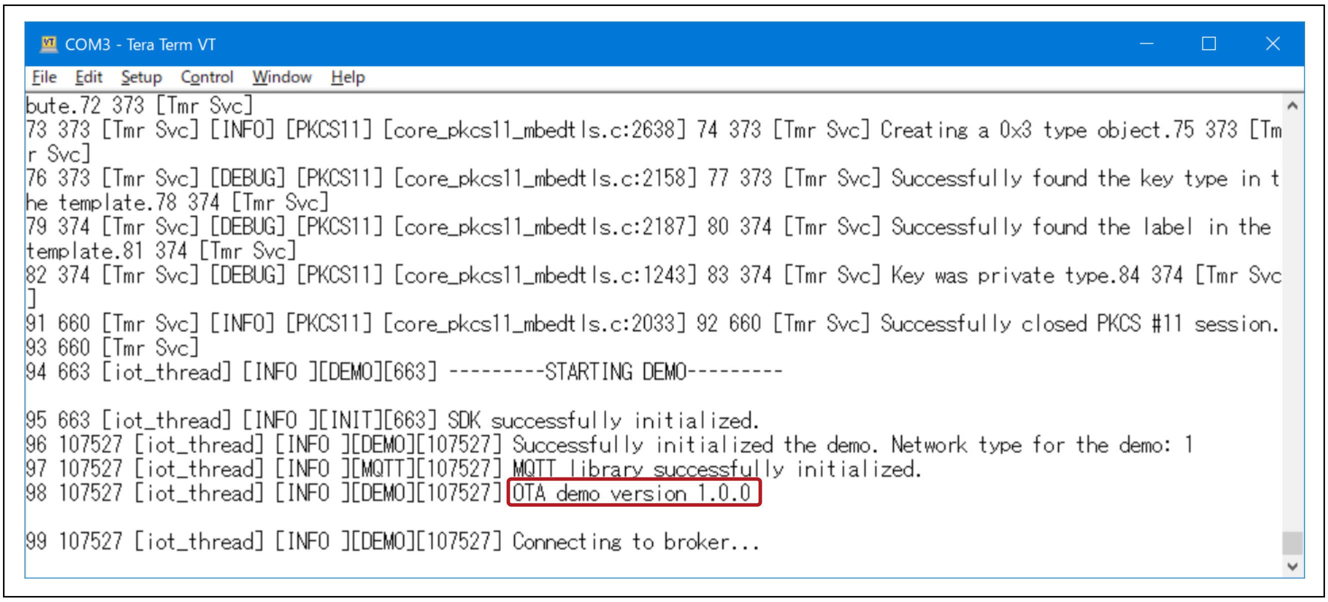


**Figure 6-5 Check Execution Result**

The Tera Term setup is shown below. If do not have Tera Term on PC, please download from https://ttssh2.osdn.jp/index.html.en.

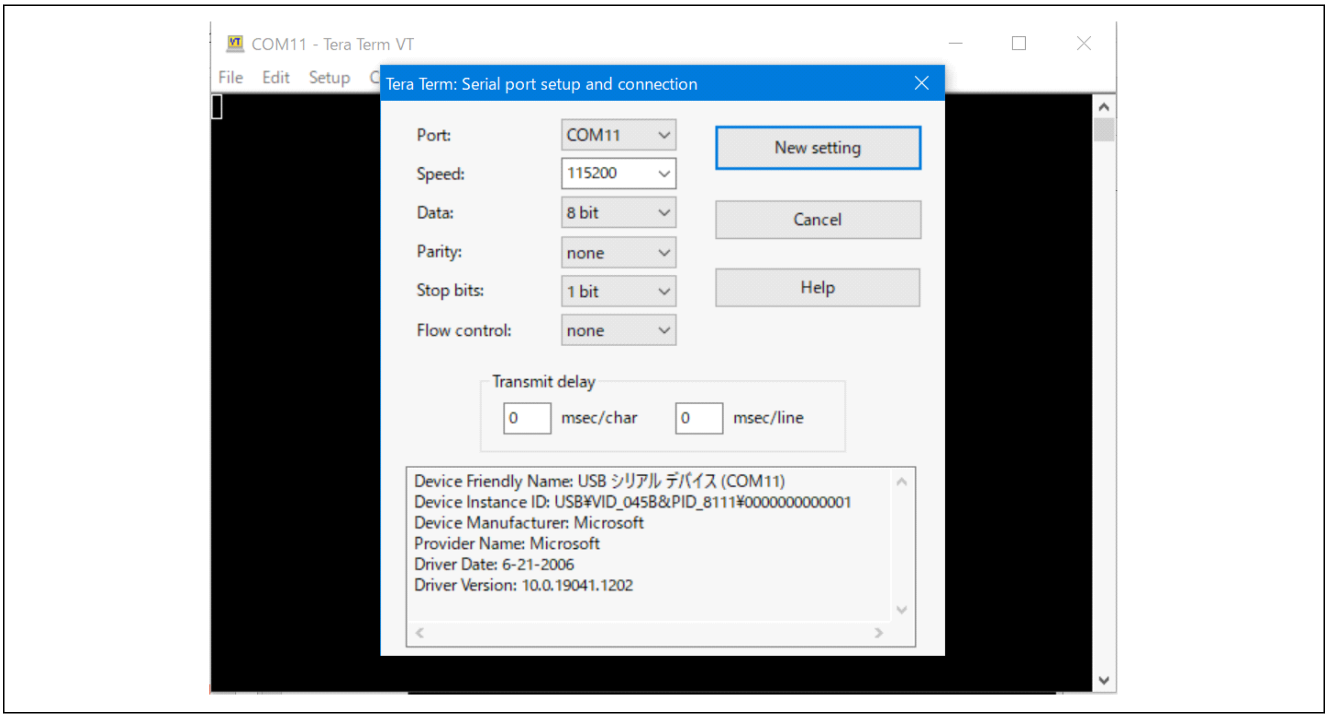


**Figure 6-6 Tera Term**

5. Check job status to be "Succeeded”.

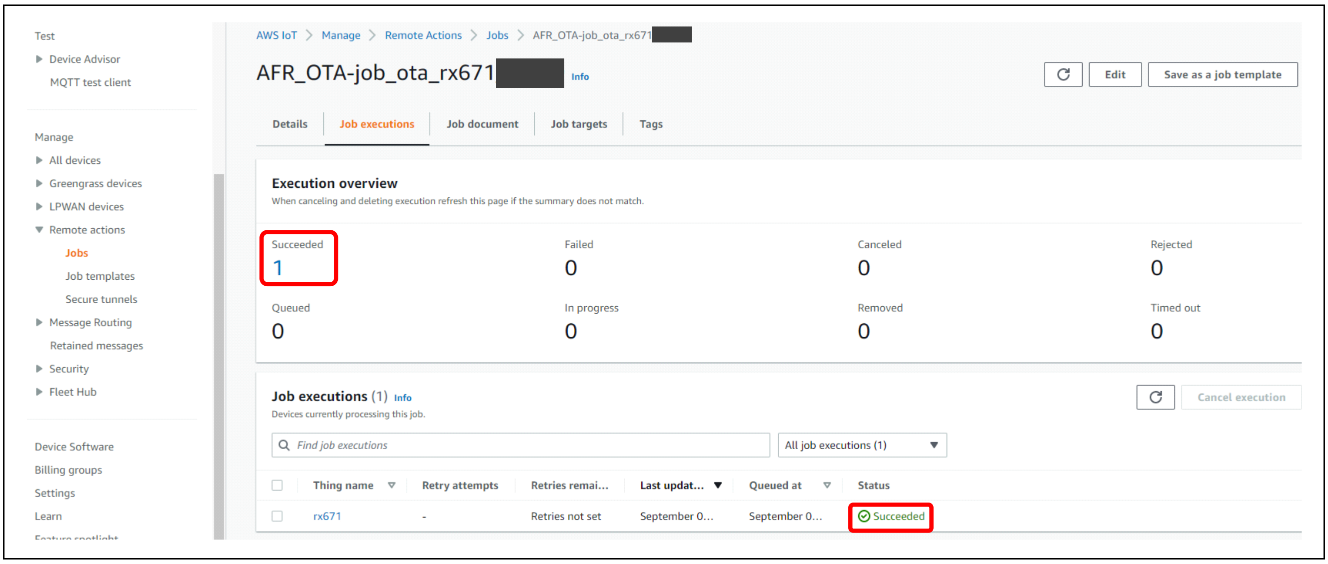


**Figure 6-7 Check Succeeded**

# 7. How to create a user program that supports OTA

This section describes how to rewrite the program from the cloud by OTA.
The program is rewritten in the background and automatically switched to the new program at the next power-on.
First, user can select the version of Amazon FreeRTOS package. The selected version will be downloaded from GitHub and imported automatically into the project. This makes it easier for the user, so that the user can focus only on Amazon FreeRTOS configuration and writing program.

## 7.1 AWS Preparation

To perform OTA from the cloud, it is necessary to prepare a cloud environment.
Use AWS as the cloud. Refer to the following for more information on preparing for AWS.
RX Family How to implement FreeRTOS OTA by using Amazon Web Services on RX65N (R01AN5549).

## 7.2 Import, configurate head file and build aws_demos and boot_loader

The figure below shows how to import Amazon FreeRTOS project:

1. Launch e² studio.
2. Select [File] → [Import…]
3. Select [Renesas GitHub FreeRTOS (with IoT libraries) Project]

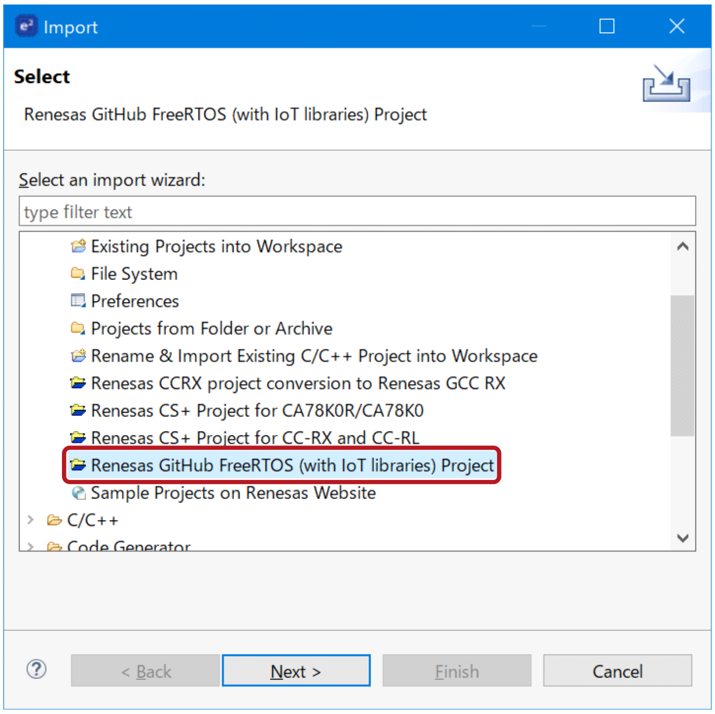


**Figure 7-1 Importing Projects**

4. Click [Check for more version…] to show the "FreeRTOS (with IoT libraries)" dialog.

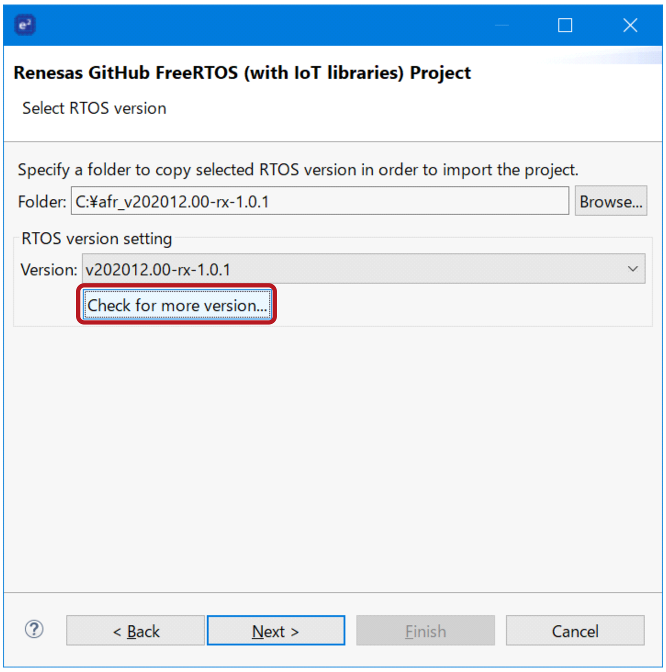


**Figure 7-2 "FreeRTOS (with IoT libraries)" dialog**

5. Select the latest version. (If the latest version is not displayed, create a new e$^2$ studio workspace)

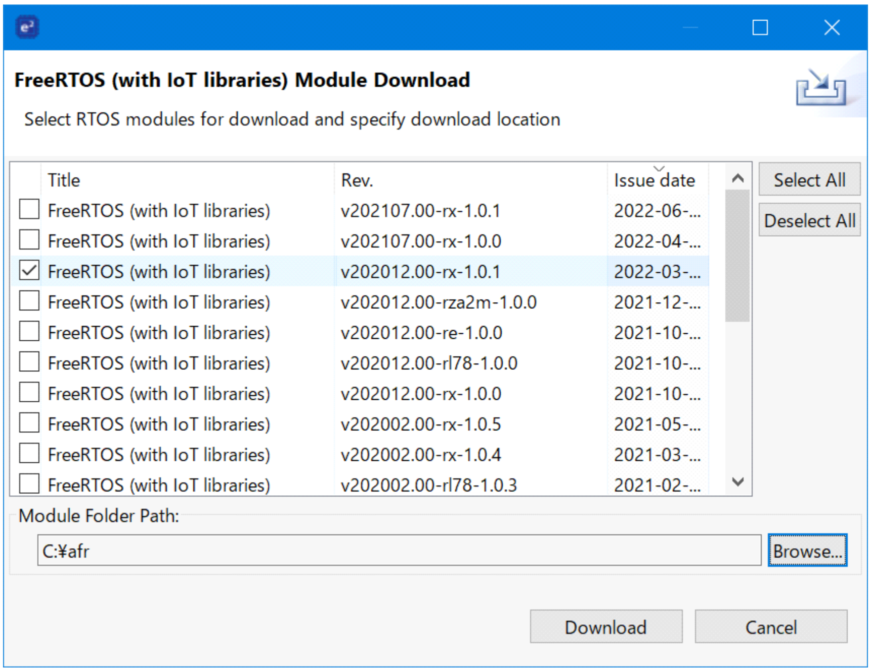


**Figure 7-3 Select OS version**

6. Agree to the end user license agreement

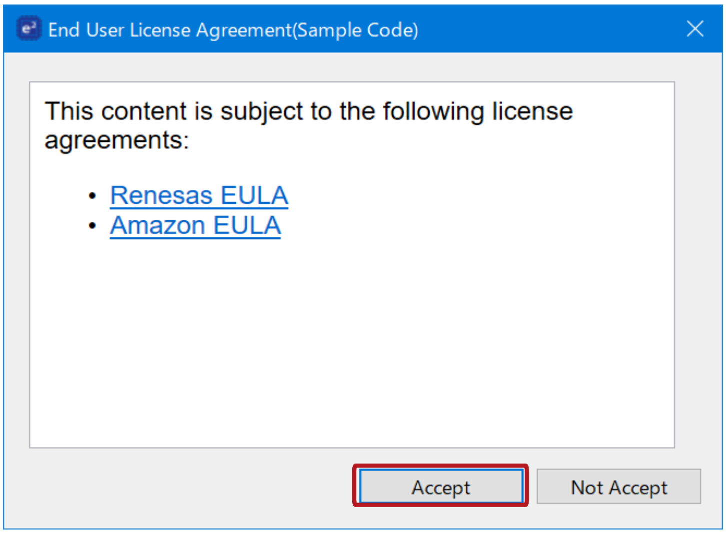


**Figure 7-4 Agree to End User License Agreement**

7. Wait for the download to complete.

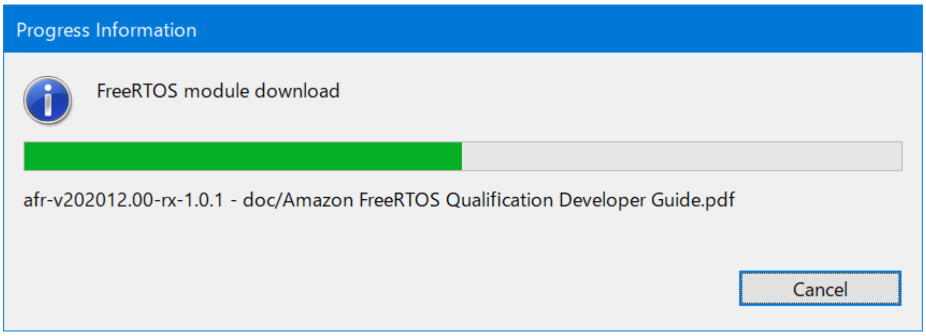


**Figure 7-5 Waiting for download**

8. Select the project to import. Select [aws_demos] and [boot_loader] project.

The aws_demos and boot_loader used in this demonstration have not been officially released in May 2023. Please contact our distributors and sales offices to request individual supplies.

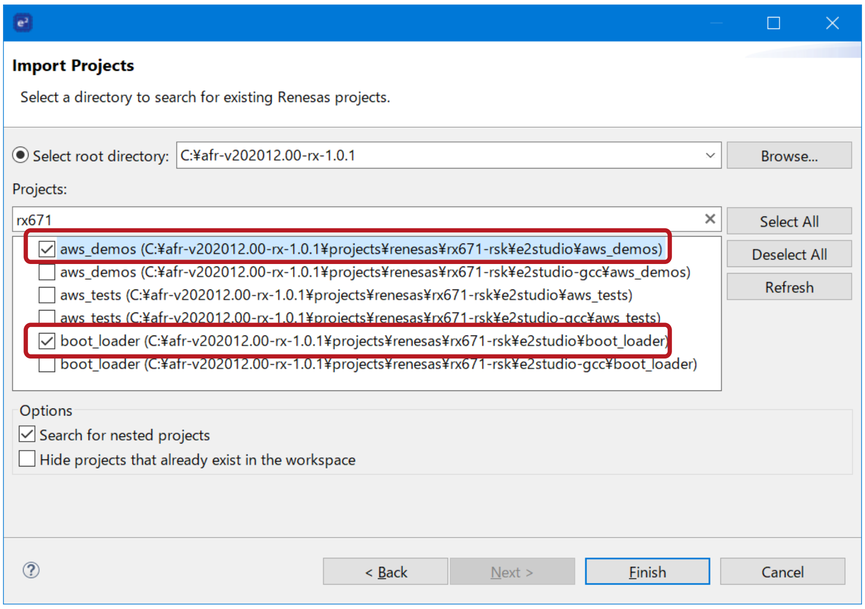


**Figure 7-6 Select a project to import**

9. Open [Project] → [Properties] → [C/C++ Build] → [Tool Chain Editor] in both projects and select "Toolchain" and "Builder" to set the toolchain. Also, select [Setting]" → [Toolchain] to set the version.

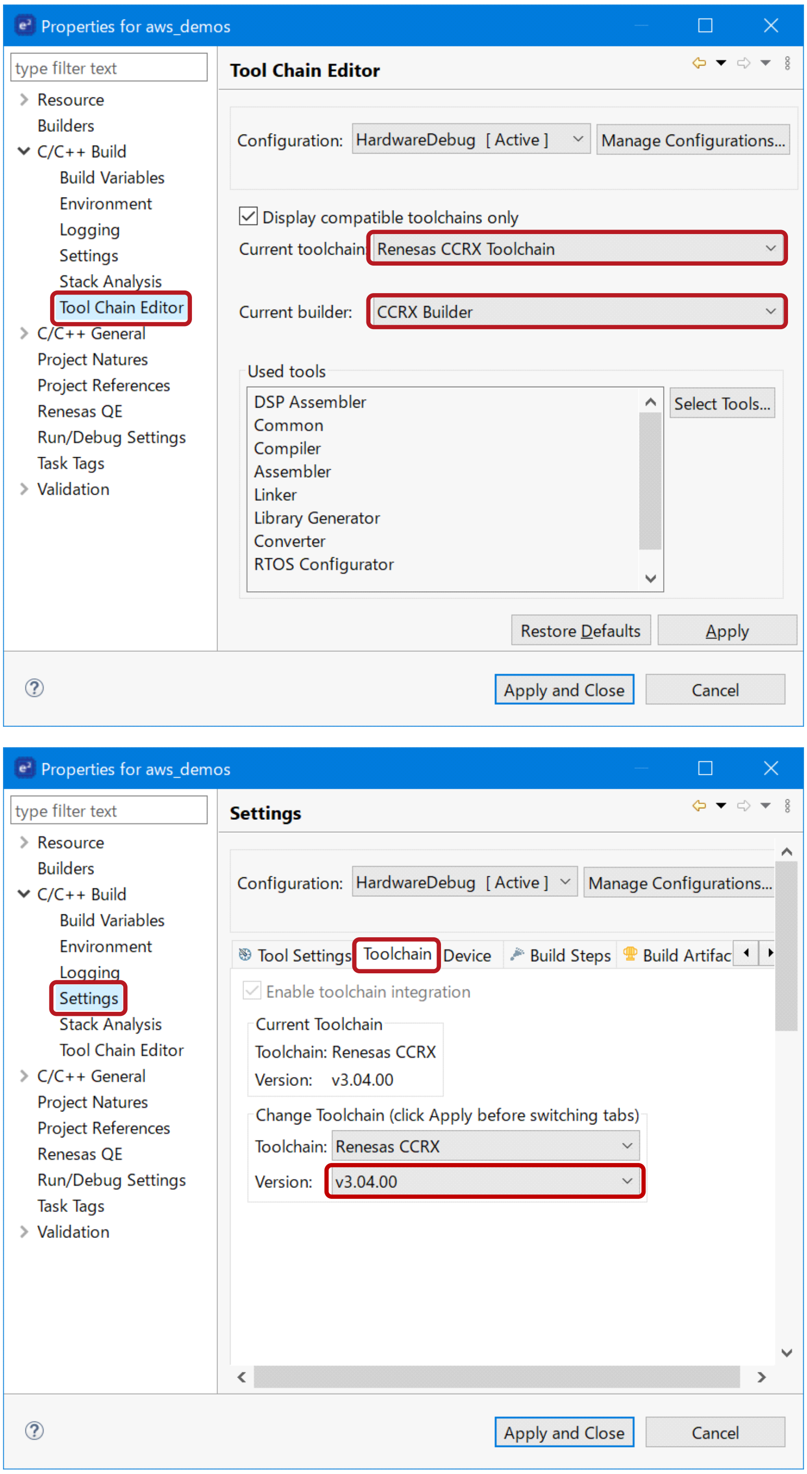


**Figure 7-7 Tool Chain and Versioning**

10. Select [Project] → [Properties] → [C/C++ Build] → [Settings] → [Converter] → [Output] and set [Motorola S format file].

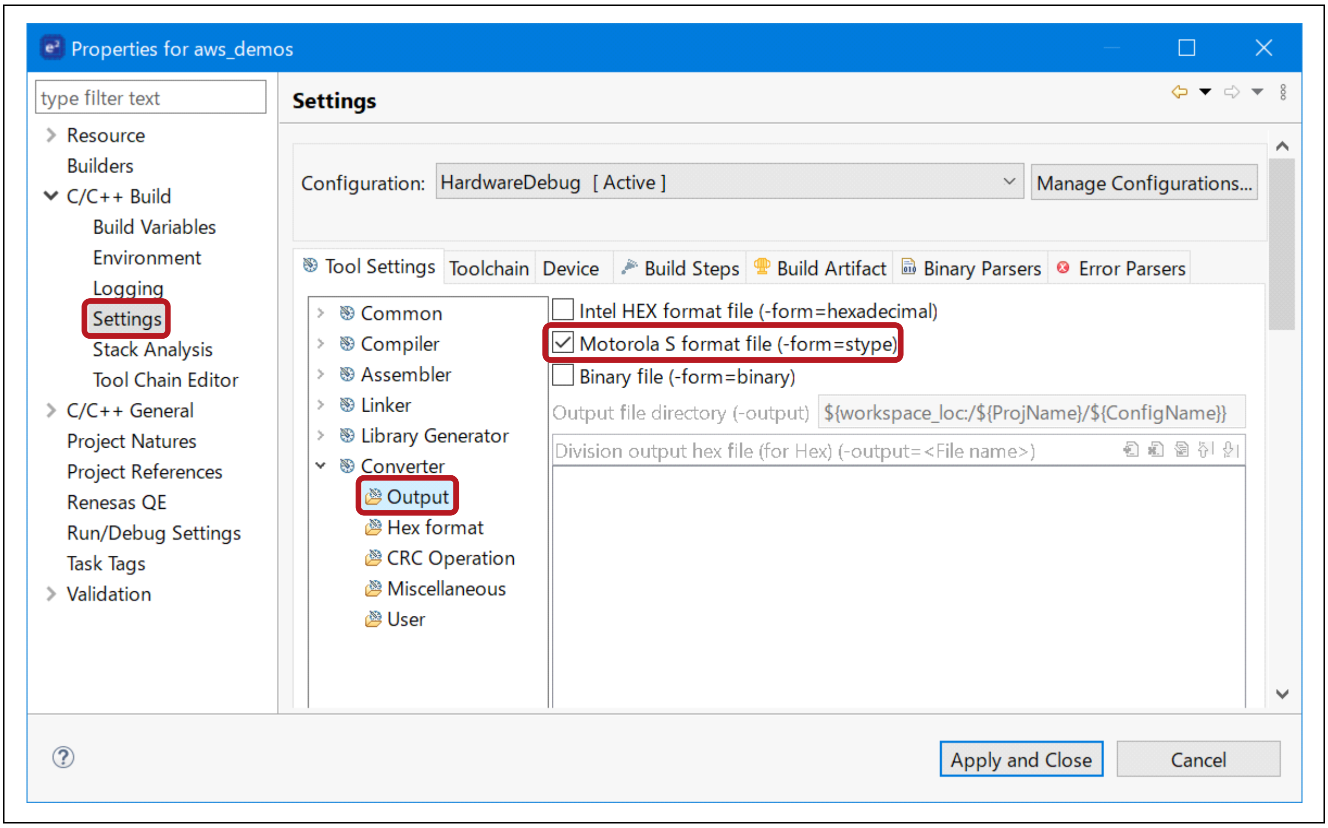


**Figure 7-8 Motorola S format File output settings**

11. Input the public key

In bootloader project, open projects\renesas\rx671-rsk\e2studio\boot_loader\src\key\code_signer_public_key.h and input public key. Please refer to “Renesas MCU Firmware Update Design Policy” section “7.3 Generating ECDSA-SHA256 Key Pairs with OpenSSL” to create public key.

When completed to create public key, build and generate the boot_loader.mot file for the boot loader project.

The aws_demos and boot_loader used in this demonstration have not been officially released in May 2023.
Please contact our distributors and sales offices to request individual supplies.

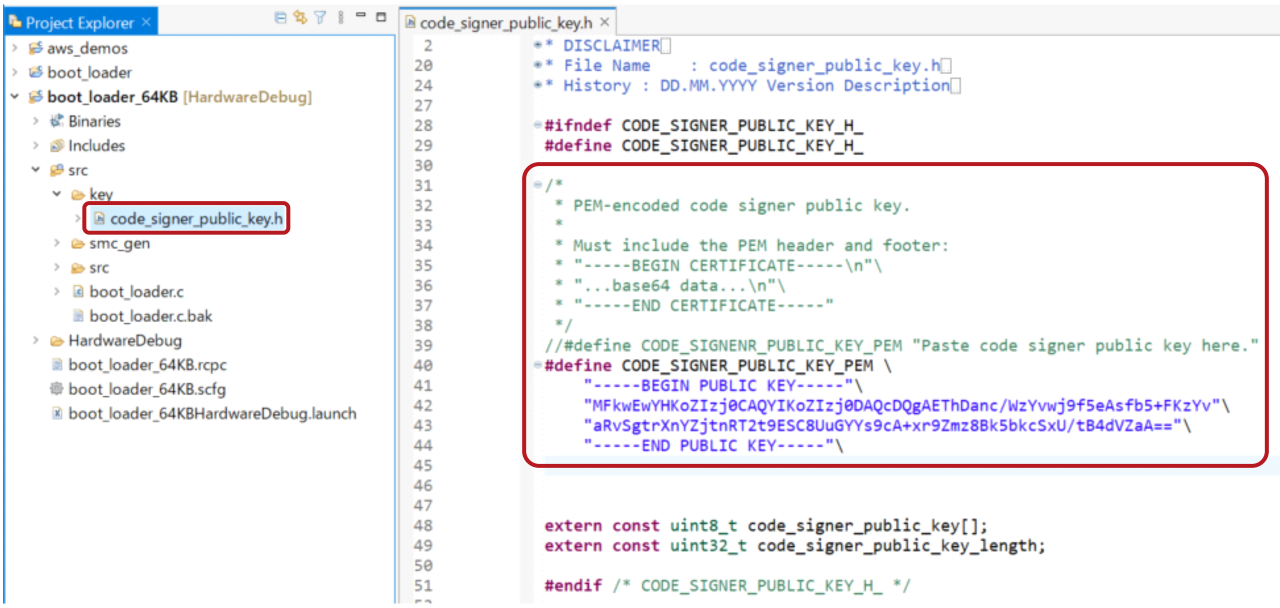


**Figure 7-9 Input the public key**

12. Open AWS IoT console
    —Browse to the AWS IoT console.
    —Select [Settings]. Make a note of the Endpoint. (Figure 7-10(e))

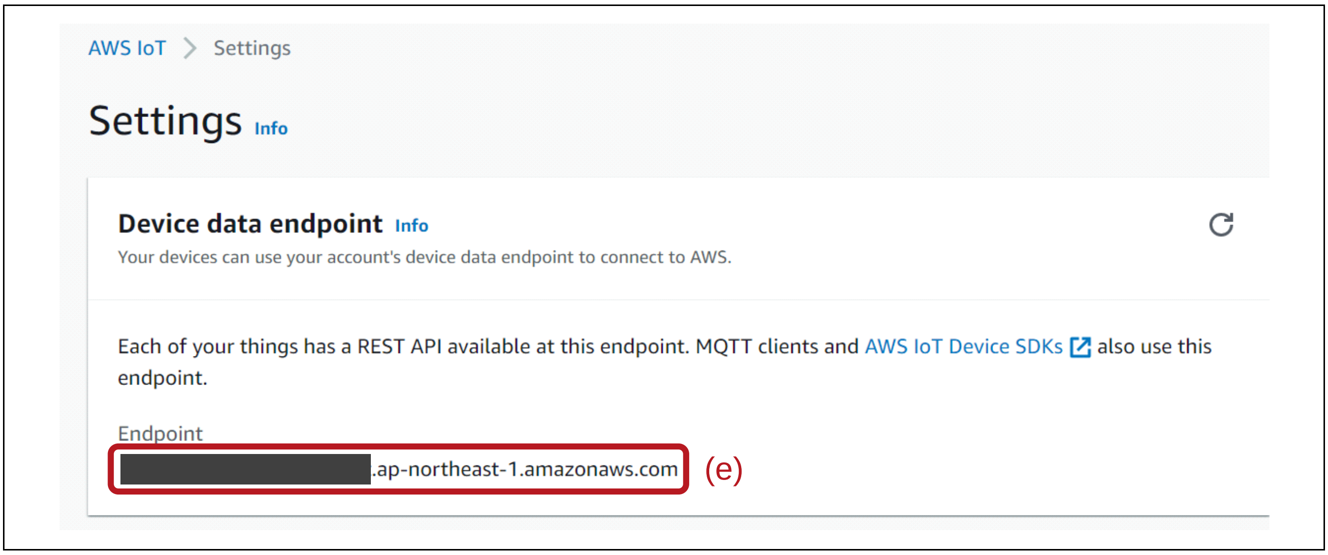


**Figure 7-10 Check AWS Endpoint**

—Select [Manage] → [Things]. Make a note of AWS IoT thing name. (Figure 7-11(f))

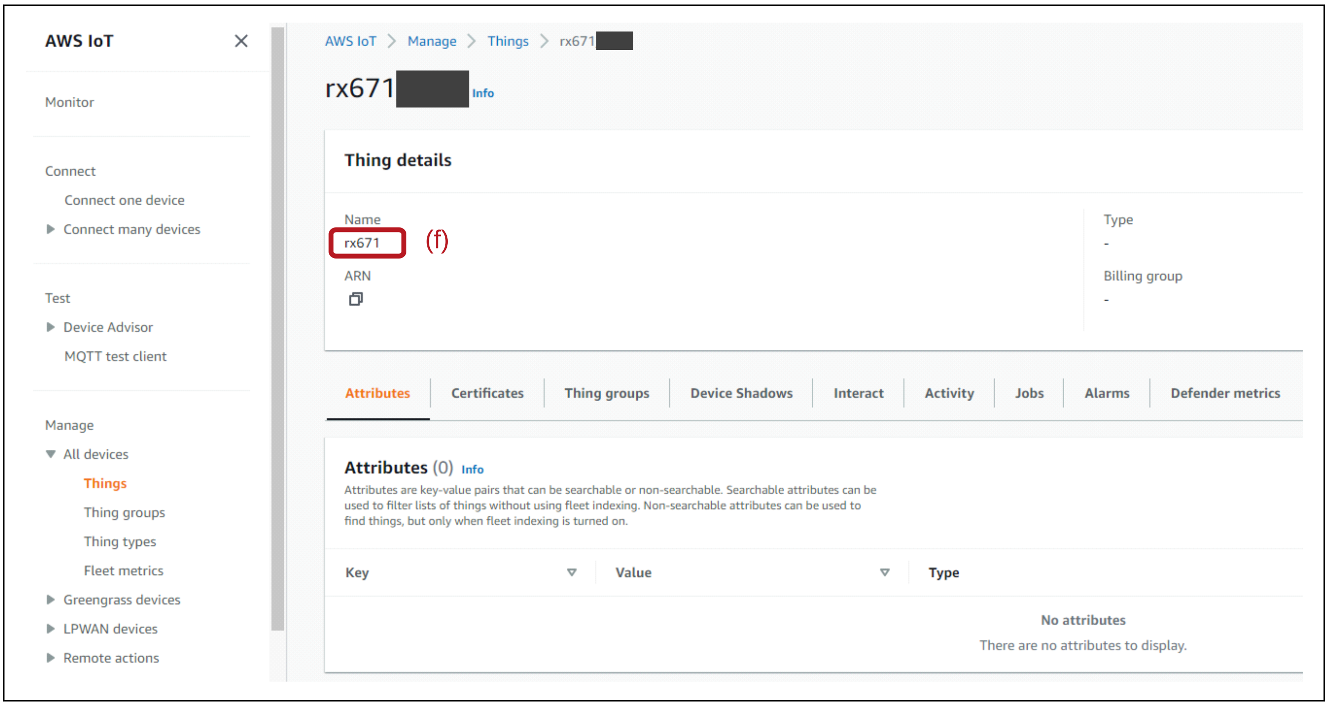


**Figure 7-11 thing name**

13. Open aws_demos project.

—Open /demos/include/aws_clientcredential.h and specify the following values

#define clientcredentialMQTT_BROKER_ENDPOINT "Figure 7-10 (e) The Endpoint"

#define clientcredentialIOT_THING_NAME "Figure 7-11 (f) thing name"

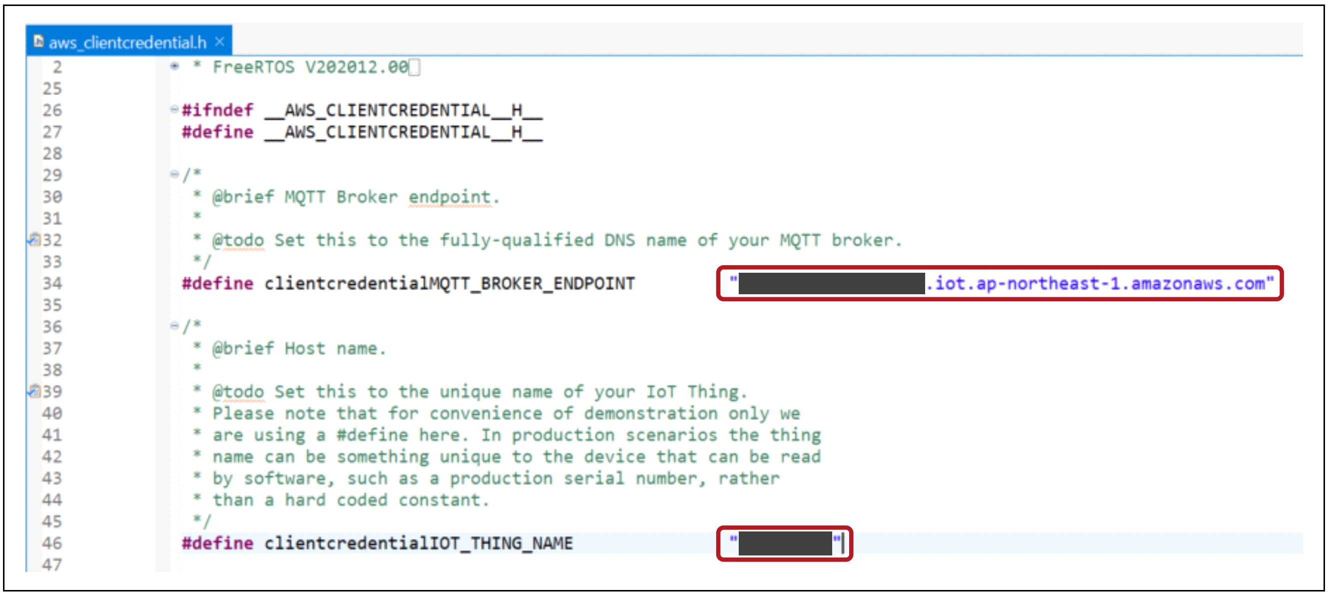


**Figure 7-12 Input the endpoint and thing name**

14. Open “Certificate Configuration Tool”

—Move to the FreeRTOS path downloaded in 7.1 step 5.

—Open [tools] → [certificate_congiguration] → CertificateConfigurator.html

—Import certificate PEM file and Private Key PEM file which were downloaded on 1.1 step (4) of "How to implement FreeRTOS OTA by using Amazon Web Services on RX65N"

—Generate awa_clientcredential_keys.h.


**Figure 7-13 Generate clientcredential key**

15. Open aws_demos project again
    - —Replace the aws_clientcredentia_keys.h generated above with the file in /demos/include/.
    - —Open /demos/include/aws_ota_codesigner_certificate.h specify values below
      signingcredentialSIGNING_CRETIFICATE_PEM [] = “xxxx”;
      “xxxx” is value from secp256r1.crt. Remember the “\” after each line of certification.
      For creating secp256r1.crt please refer to “How to implement FreeRTOS OTA by using Amazon web Services on RX65N” section “7.3 Generating ECDSA-SHA256 Key Pairs with OpenSSL”.

aws_ota_codesigner_certificate.h

```
 * FreeRTOS V202012.00

#ifndef __AWS_CODESIGN_KEYS__H__
#define __AWS_CODESIGN_KEYS__H__

/*
 * PEM-encoded code signer certificate
 *
 * Must include the PEM header and footer:
 * "-----BEGIN CERTIFICATE-----\n"
 * "...base64 data...\n"
 * "-----END CERTIFICATE-----\n";
 */
static const char signingcredentialSIGNING_CERTIFICATE_PEM[] =
"-----BEGIN CERTIFICATE-----\n"\
[redacted]
"KVBN+xcN\n"\
"-----END CERTIFICATE-----\n";

#endif
```

**Figure 7-14 Input clientcredential**

## 7.3 Install the initial version of firmware

1. Check the FreeRTOSApplicationConfig.h setting.

FreeRTOSApplicationConfig.h

```c
#ifndef FREERTOS_APPLICATION_CONFIG_H
#define FREERTOS_APPLICATION_CONFIG_H

#define OTA             (Used)

#define Used            (1)
#define Unused          (0)

#define CONNECTION              (WIFI)

#define ETHER            (1)
#define WIFI          (0)

#if (CONNECTION == 1)
#error "Connection type ETHER not supported"
#endif

#endif
```

**Figure 7-15 Check Setup**

2. Open amazon-freertos/demos/include/ aws_application_version.h, set initial version of firmware to 0.90.

aws_application_version.h

```c
 * FreeRTOS V202012.00

#ifndef _AWS_APPLICATION_VERSION_H_
#define _AWS_APPLICATION_VERSION_H_

#include "iot_appversion32.h"
extern const AppVersion32_t xAppFirmwareVersion;

#define APP_VERSION_MAJOR     (0U)
#define APP_VERSION_MINOR     (9U)
#define APP_VERSION_BUILD     (0U)

#endif
```

**Figure 7-16 Firmware initial version definition**

3. Open Section Viewer by selecting [Project] → [Properties] → [C / C ++ Build] → [Settings] → [Tool Settings] tab → [Linker] → [Section] → [...] and change section of aws_demos as following picture.

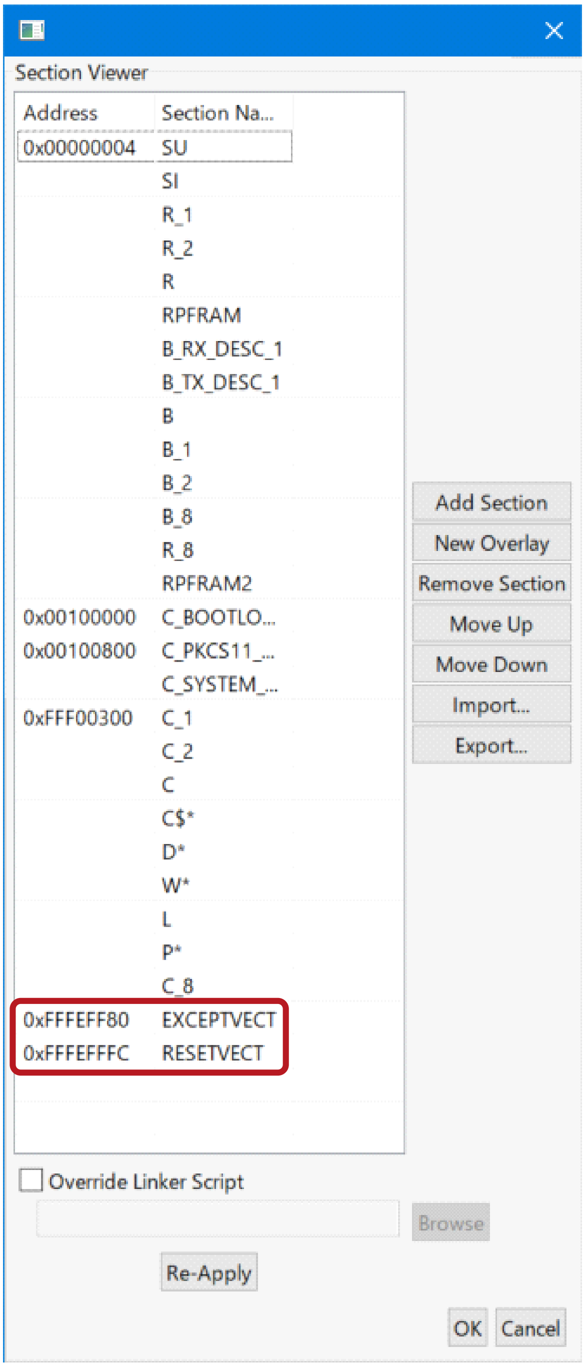


**Figure 7-17 Section Settings**

4. Build to create aws_demos.mot file.

5. Create userprog.mot from Renesas Secure Flash Programmer.
   userprog.mot is a combination of aws_demos.mot and boot_loader.mot. Users can flash this file to RX671 PoC to install initial firmware.
   Download Renesas Secure Flash Programmer release 1.0.1 and open Renesas Secure Flash Programmer.exe. Also downloads other files.
   —Select [Initial Firm] tab and then set parameters as following picture.
   - Private Key Path : location to secp256r1.privatekey
   - Boot Loader File Path : location to boot_loader.mot
     (¥projects¥renesas¥rx671-rsk¥e2studio¥boot_loader¥HardwareDebug)
   - Bank 0 User Program File Path : location to aws_demos.mot
     (¥projects¥renesas¥rx671-rsk¥e2studio¥aws_demos¥HardwareDebug)

   —Select [Generate] to generate userprog.mot and save it in the init_firmware folder.
   Check Generate succeeded is displayed.

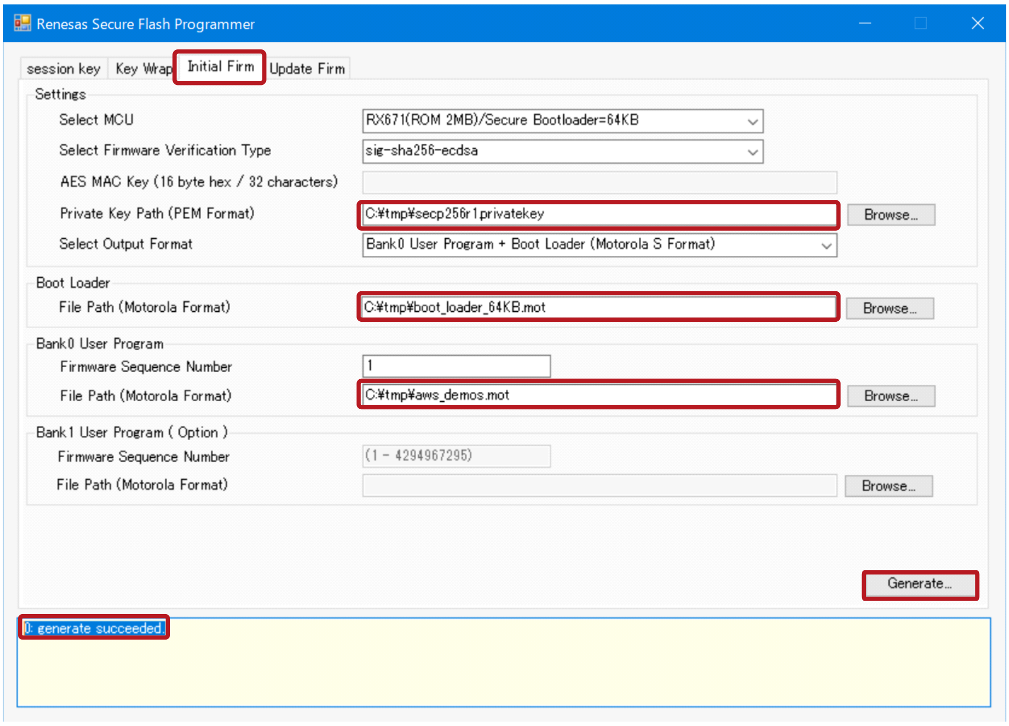


**Figure 7-18 Generate userprog.mot**

6. Erase the flash ROM of the RX671 PoC.

— Download the latest version of Renesas Flash Programmer below.
https://www.renesas.com/rfp

— Open the following project in Renesas Flash Programmer.
¥vendors¥renesas¥rx_mcu_boards¥boards¥rx671-rsk¥aws_demos¥flash_project
¥erase_from_bank¥erase.rpj

— Select [Operation] tab and click [Start] to erase the flash ROM.

要修正

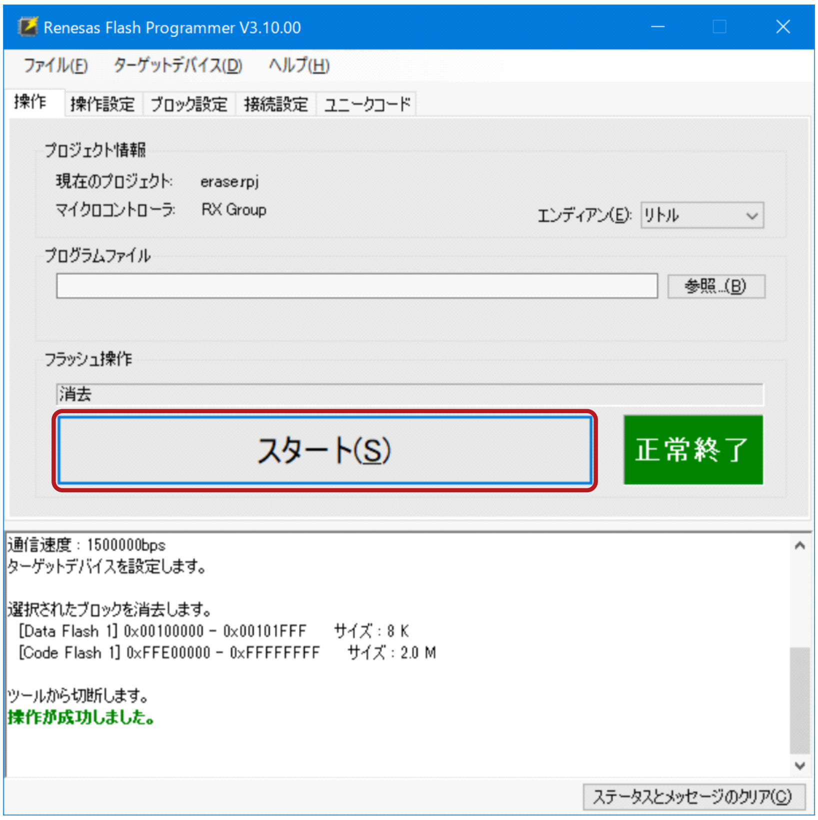


**Figure 7-19 Flash ROM erase**

7. Write initial firmware on RX671 PoC.

—Create a new project with Renesas Flash Programmer (Ex : flash_project.rpj)

—Select [Operation] tab and set userprog.mot stored in the init_firmware folder of the Program File.

—Click [Start].

要修正

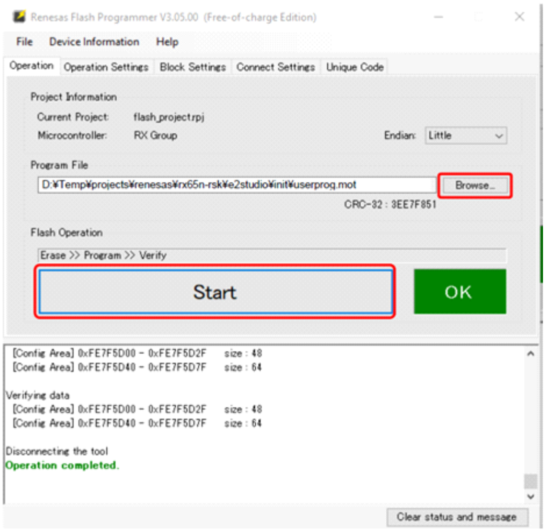


**Figure 7-20 Writing initial firmware**

8. Open Tera Term

The Tera Term setup is shown below.

要修正

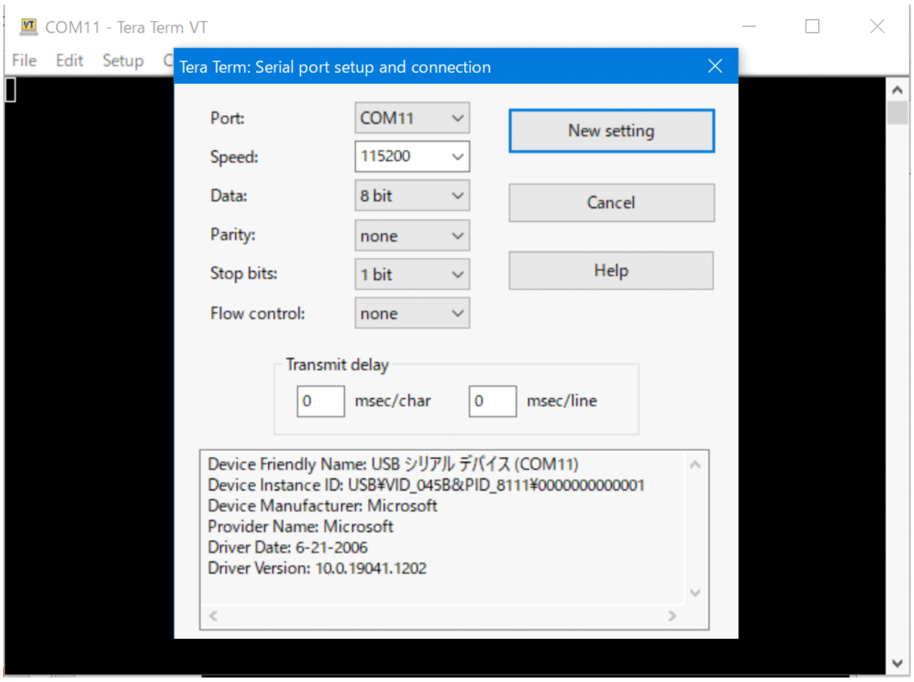


**Figure 7-21 Tera Term**

Version 0.90 (initial version) is installed in RX671 PoC. RX671 PoC is now ready to receive OTA updates. The output log is shown below.a

```
--------------------------------------------------
RX671 secure boot program
--------------------------------------------------
Checking data flash ROM status.
Loading user code signer public key: not found.
provision the user code signer public key: OK.
Checking code flash ROM status.
bank 0 status = 0xfc [LIFECYCLE_STATE_INITIAL_FIRM_INSTALLED]
bank 1 status = 0xff [LIFECYCLE_STATE_BLANK]
bank info = 1. (start bank = 0)
started 10us software timer using CMT channel 0.
integrity check scheme = sig-sha256-ecdsa
bank0(execute area) on code flash integrity check...OK
erase bank1 secure boot mirror area...OK
OK
copy secure boot (part2) from bank0 to bank1...OK
jump to user program
0 83 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:449] 1 83 [Tmr Svc] PKCS #11
module was successfully initialized.2 83 [Tmr Svc]
3 83 [Tmr Svc] [INFO] [PKCS11] [core_pkcs11_mbedtls.c:1504] 4 83 [Tmr Svc] PKCS #11
successfully initialized.5 83 [Tmr Svc]
6 83 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1717] 7 83 [Tmr Svc] Successfully
Returned a PKCS #11 slot with ID 1 with a count of 1.8 83 [Tmr Svc]
9 83 [Tmr Svc] [WARN] [PKCS11] [core_pkcs11_mbedtls.c:1749] 10 83 [Tmr Svc]
C_GetTokenInfo is not implemented.11 83 [Tmr Svc]
12 83 [Tmr Svc] [WARN] [PKCS11] [core_pkcs11_mbedtls.c:1839] 13 83 [Tmr Svc] C_InitToken
is not implemented.14 83 [Tmr Svc]
70 373 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:471] 71 373 [Tmr Svc]
Successfully found object class attribute.72 373 [Tmr Svc]
```

```
73 373 [Tmr Svc] [INFO] [PKCS11] [core_pkcs11_mbedtls.c:2638] 74 373 [Tmr Svc] Creating a
0x3 type object.75 373 [Tmr Svc]
76 373 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:2158] 77 373 [Tmr Svc]
Successfully found the key type in the template.78 374 [Tmr Svc]
79 374 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:2187] 80 374 [Tmr Svc]
Successfully found the label in the template.81 374 [Tmr Svc]
82 374 [Tmr Svc] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1243] 83 374 [Tmr Svc] Key was
private type.84 374 [Tmr Svc]
91 660 [Tmr Svc] [INFO] [PKCS11] [core_pkcs11_mbedtls.c:2033] 92 660 [Tmr Svc]
Successfully closed PKCS #11 session.93 660 [Tmr Svc]
94 663 [iot_thread] [INFO ][DEMO][663] ---------STARTING DEMO---------

95 663 [iot_thread] [INFO ][INIT][663] SDK successfully initialized.
96 107527 [iot_thread] [INFO ][DEMO][107527] Successfully initialized the demo. Network
type for the demo: 1
97 107527 [iot_thread] [INFO ][MQTT][107527] MQTT library successfully initialized.
98 107527 [iot_thread] [INFO ][DEMO][107527] OTA demo version 0.9.0

99 107527 [iot_thread] [INFO ][DEMO][107527] Connecting to broker...

100 107527 [iot_thread] [INFO ][DEMO][107527] MQTT demo client identifier is rx671_POC
(length 9).
101 109439 [iot_thread] [WARN] [PKCS11] [core_pkcs11_mbedtls.c:1499] 102 109439
[iot_thread] Failed to initialize PKCS #11. PKCS #11 was already initialized.103 109439
[iot_thread]
104 109439 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1717] 105 109439
[iot_thread] Successfully Returned a PKCS #11 slot with ID 1 with a count of 1.106 109439
[iot_thread]
107 109439 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1953] 108 109439
[iot_thread] Assigned a 0x2 Type Session.109 109439 [iot_thread]
110 109439 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1964] 111 109439
[iot_thread] Assigned Mechanisms to no operation in progress.112 109439 [iot_thread]
113 109439 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1980] 114 109439
[iot_thread] Current session count at 0115 109439 [iot_thread]
167 111156 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:1073] 168 111156
[iot_thread] Found object in list by handle.169 111156 [iot_thread]
170 111156 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:3780] 171 111157
[iot_thread] Successfully started sign operation.172 111157 [iot_thread]
173 112138 [iot_thread] [DEBUG] [PKCS11] [core_pkcs11_mbedtls.c:3966] 174 112138
[iot_thread] Ended Sign operation.175 112138 [iot_thread]
176 112223 [iot_thread] [INFO ][MQTT][112223] Establishing new MQTT connection.
177 112223 [iot_thread] [INFO ][MQTT][112223] (MQTT connection 23f18, CONNECT operation
240b8) Waiting for operation completion.
178 112351 [NetRecv] [INFO] [MQTT] [core_mqtt_serializer.c:970] 179 112351 [NetRecv]
CONNACK session present bit not set.180 112351 [NetRecv]
181 112351 [NetRecv] [INFO] [MQTT] [core_mqtt_serializer.c:912] 182 112351 [NetRecv]
Connection accepted.183 112351 [NetRecv] no
184 112351 [iot_thread] [INFO ][MQTT][112351] (MQTT connection 23f18, CONNECT operation
240b8) Wait complete with result SUCCESS.
185 112351 [iot_thread] [INFO ][MQTT][112351] New MQTT connection 6a54 established.
186 112353 [iot_thread] [OTA_AgentInit_internal] OTA Task is Ready.
187 112353 [OTA Agent T] [prvPAL_GetPlatformImageState] is called.
188 112353 [OTA Agent T] Function call: prvPAL_GetPlatformImageState: [2]
189 112353 [OTA Agent T] [prvExecuteHandler] Called handler. Current State [Ready] Event
[Start] New state [RequestingJob]
190 112358 [OTA Agent T] [INFO ][MQTT][112358] (MQTT connection 23f18) SUBSCRIBE
operation scheduled.
191 112358 [OTA Agent T] [INFO ][MQTT][112358] (MQTT connection 23f18, SUBSCRIBE
operation 2a960) Waiting for operation completion.
192 112473 [OTA Agent T] [INFO ][MQTT][112473] (MQTT connection 23f18, SUBSCRIBE
operation 2a960) Wait complete with result SUCCESS.
193 112473 [OTA Agent T] [prvSubscribeToJobNotificationTopics] OK:
$aws/things/rx671_POC/jobs/$next/get/accepted
194 112478 [OTA Agent T] [INFO ][MQTT][112478] (MQTT connection 23f18) SUBSCRIBE
operation scheduled.
195 112478 [OTA Agent T] [INFO ][MQTT][112478] (MQTT connection 23f18, SUBSCRIBE
operation 241f8) Waiting for operation completion.
```

```
196 112585 [OTA Agent T] [INFO ][MQTT][112585] (MQTT connection 23f18, SUBSCRIBE
operation 241f8) Wait complete with result SUCCESS.
197 112585 [OTA Agent T] [prvSubscribeToJobNotificationTopics] OK:
$aws/things/rx671_POC/jobs/notify-next
198 112585 [OTA Agent T] [prvRequestJob_Mqtt] Request #0
199 112594 [OTA Agent T] [INFO ][MQTT][112594] (MQTT connection 23f18) MQTT PUBLISH
operation queued.
200 112594 [OTA Agent T] [INFO ][MQTT][112594] (MQTT connection 23f18, PUBLISH operation
241f8) Waiting for operation completion.
201 112670 [OTA Agent T] [INFO ][MQTT][112670] (MQTT connection 23f18, PUBLISH operation
241f8) Wait complete with result SUCCESS.
202 112670 [OTA Agent T] [prvExecuteHandler] Called handler. Current State
[RequestingJob] Event [RequestJobDocument] New state [WaitingForJob]
203 112672 [OTA Agent T] [prvParseJSONbyModel] Extracted parameter [ clientToken:
0:rx671_POC ]
204 112672 [OTA Agent T] [prvParseJSONbyModel] Extracted parameter [ timestamp:
1662611090 ]
205 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: execution
206 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: jobId
207 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: jobDocument
208 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: afr_ota
209 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: protocols
210 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: files
211 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: filepath
212 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: filesize
213 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: fileid
214 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: certfile
215 112672 [OTA Agent T] [prvParseJSONbyModel] parameter not present: sig-sha256-ecdsa
216 112672 [OTA Agent T] [prvParseJobDoc] No active jobs available in the service for
execution.
217 112674 [OTA Agent T] [prvParseJobDoc] Ignoring job without ID.
222 113353 [iot_thread] [INFO ][DEMO][113353] State: Ready  Received: 1   Queued: 0
Processed: 0   Dropped: 0

223 114353 [iot_thread] [INFO ][DEMO][114353] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0

224 115353 [iot_thread] [INFO ][DEMO][115353] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0

225 116353 [iot_thread] [INFO ][DEMO][116353] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0

226 117353 [iot_thread] [INFO ][DEMO][117353] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0

227 118353 [iot_thread] [INFO ][DEMO][118353] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0
```

## 8. Restriction

This section describes restriction for this application note.

- FreeRTOS OTA programs with big endian operate abnormally.
  Build and operate programs with little endian.

## 9. Reference Documents


- RX671Group User’s Manual: Hardware (R01UH0905)
- Renesas Starter Kit+ for RX671User’s Manual (R20UT4879)
- RX Family Using QE and FIT to Develop Capacitive Touch Applications (R01AN4516)
- RX Family QE for Display GUI Display Application Development Guide using Serial Connection LCD (R20AN0688)
- Renesas MCU Firmware Update Design Policy (R01AN5548)
- How to implement FreeRTOS OTA by using Amazon Web Serviceson RX65N (R01AN5549)


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

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Apr.24.23 | - | First edition |
| | | | |

# General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

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

1. Precaution against Electrostatic Discharge (ESD)

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

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

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

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

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

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

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

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

## Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

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

(Rev.5.0-1 October 2020)


## Corporate Headquarters

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


## Trademarks

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

## Contact information

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