Arm® 926-EJS

32-bit Microprocessor

NuMaker-Thermostat-N9H20
User Manual

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

Table of Contents

1 OVERVIEW

NuMaker-Thermostat-N9H20 is a GUI reference implementation for thermostat HMI.

This reference implemenation is based on the Nuvoton N9H20 series general-purpose microprocessor N9H20K3 (with 8MB DDR) or N9H20K5 (with 32MB DDR) to implement thermostat HMI by emWin GUI library. In addition to the Thermostat HMI, it also supports Modbus Master RTU protocol to communicate and control Modbus Slave devices.

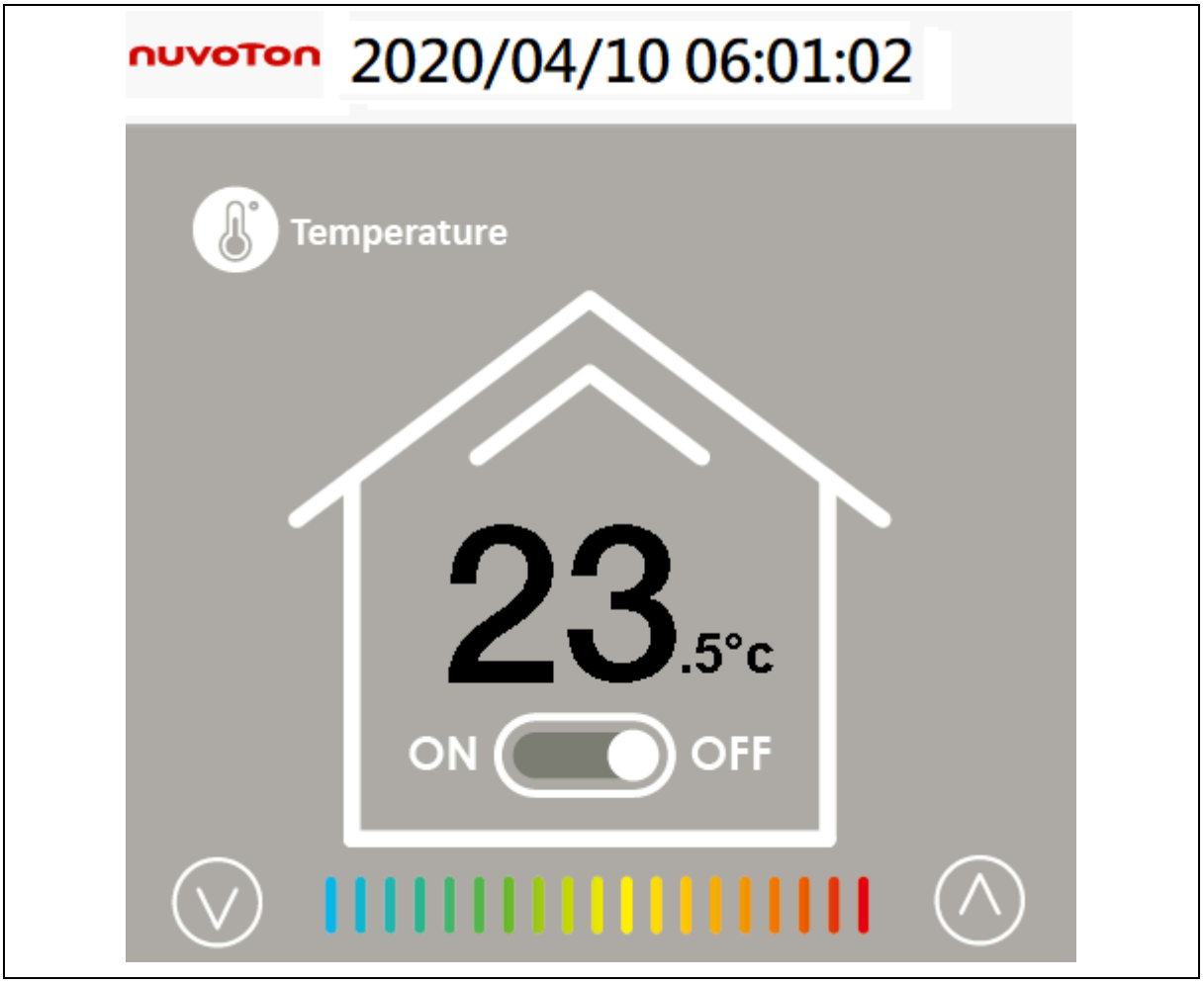

Figure 1-1 Thermostat HMI Temperature Control Menu

2 FEATURES

2.1 Thermostat HMI Features

- Supports SEGGER licensed emWin GUI library
- Supports capacitive touch via I²C interface
- Supports high quality and contrast IPS LCD panel with resolution up to 480 x 480
- Supports simulated RS485 via high speed UART
- Supports Modbus Master RTU protocol
- Supports motion effect (slide to left or right) of three menus, as shown below.

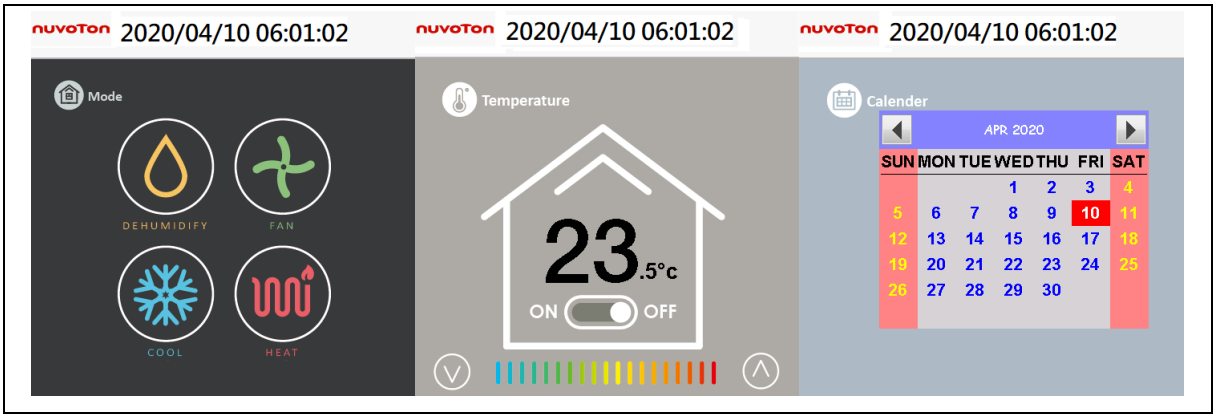

Figure 2-1 Three Sliding Menus (Mode, Temperature and Calendar)

3 INSTALLATION ENVIRONMENT

3.1 System Requirements

- KEIL MDK V5.xx and above with Plus or Professional edition license or NuEclipse (GCC)
- Nuvoton N9H20K3 or N9H20K5 480 x 480 thermostat demo board
- Modbus Slave device M487 (optional)
- Modbus Slave device PZEM-003 (optional)

3.2 Installing N9H20

First, download the latest N9H20 BSP from https://github.com/OpenNuvoton/N9H20_emWin_NonOS, and unzip *N9H20_emWin_NonOS-master.zip* to a working folder, e.g., unzip it to the path *C:\N9H20*, where *N9H20* is the working folder.

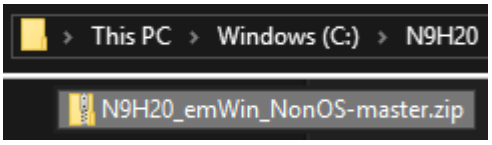

Figure 3-1 N9H20 BSP File Name and Working Folder

The detailed information of N9H20 BSP *N9H20 Readme* and emWin library *UM03001_emWin* can be found in *N9H20_emWin_NonOS-master* and *N9H20_emWin_NonOS-master\BSP\ThirdParty\emWin\Doc* respectively.

3.3 Installing Thermostat HMI

First, download the latest *SW_N9H20_Thermostat_HMI_Example_V1.5.zip* from https://www.nuvoton.com/resource-download.jsp?tp_GUID=EC0120200305164244 and unzip to the BSP sample path *N9H20_emWin_NonOS-master\BSP\SampleCode\emWin*.

Then, open thermostat project *ThermostatDemo.uvproj* under *N9H20_emWin_NonOS-master\BSP\SampleCode\emWin\ThermostatDemo\KEIL* and start compiling. The executable binary *conprog.bin* is under *N9H20_emWin_NonOS-master\BSP\SampleCode\emWin\ThermostatDemo\Bin*.

Next, connect USB cable between computer and N9H20 and power on. Then, copy *conprog.bin* to NAND1-1 USB disk. Finally, remove USB disk safely and reboot N9H20.

4 FOLDER STRUCTURE

4.1 Code Folder Structure

The content of *SW_N9H20_Thermostat_HMI_Example_V1.5.zip* is described as follows.

Folder	Description
ThermostatDemo	Base folder ● *main.c* is platform related initilizations
Application	Thermostat code and image folder, image folder will describe in the next chapter ● *Thermostat.c* is entry point of thermostat GUI
Bin	Pre-built binaries folder ● *conprog.bin* is Thermostat execution file ● *N9H20K5_NVT_NAND_D395T9375V0_480x480.bin* is NAND NVTLoader for 480 x 480 ● *PACKET_RGB565_size480x480.bin* is RGB565 logo for 480 x 480 LCD ● *UART_RS485_ID_2.bin* is M487 Modbus Slave execution file (optional)
Root	Root folder ● *Changelog* is Thermostat reference code change history
GCC	Eclipse project folder
KEIL	Arm Keil MDK project folder
LCD	Display and user-defined emWin setting folder ● *GUIConf2.c* is emWin memory pool setting file ● *LCDConf2.c* is emWin display and multiple buffers driver ● *N9H20_EFFECT* is platform effect library ● *N9H20_VPOST_D395T9375V0_480x480* is 480 x 480 display driver
ModBus_Master	Modbus Master folder ● *ModbusMaster.c* is Modbus Master source code ● *RS485_UART.c* is N9H20 UART + RS485 source code
Touch	Touch folder ● *FT6336.h* is FT6336 capacitive touch driver ● *N9H20TouchPanel.c* is capacitive touch driver for FT6336 ● *N9H20TouchPanel.h* is touch valid range definition

Table 4-1 Thermostat HMI Folder Structure

4.2 Image Resource Folder Structure

The *Application* folder contains image and converted c array.

Folder	Description
Common	Common bitmap ● *off1* is button off ● *on1* is button on ● *onoffselect1* is text on/off
Heading	Heading (title bar) bitmap ● *logo* is company logo
Menu1	Menu 1 bitmap ● *cool1* is background ● *cool1disable1* is button disable ● *dehumidify1* is background ● *dehumidify1disable1* is button disable ● *fan1* is background ● *fan1disable1* is button disable ● *heat1* is background ● *heat1disable1* is button disable ● *menu1* is background ● *menu1back1* is button back ● *menu1down1* is button down ● *menu1up1* is button up
Menu2	Menu 2 bitmap ● *menu2* is background ● *menu2back1* is button back ● *menu2down1* is button down ● *menu2up1* is button up
Menu3	Menu 3 bitmap ● *menu3* is background ● *menu3back1* is button back

Table 4-2 Thermostat HMI Images Folder Structure

5 DESIGN GUIDE

The Thermostat reference implementation guide assumes that you already have a mature knowledge of the following:

- IDE operation for editing and compiling
- The C programming language, how to use linker and C compiler
- The N9H20 Non-OS BSP programming knowledge
- The basic emWin programming knowledge

5.1 Motion Control

The NuMaker-Thermostat-N9H20 utilizes 480 x 480 LCM to display for thermostat and has three menus for sliding motion effects. The motion support needs to be enabled before it can be used.

```
// 1 or enable; 0 for disable motion effect

WM_MOTION_Enable(1);
```

You can achieve horizontal movability for a window that can be created with the creation flag called WM_CF_MOTION_X.

```
// Create window with motion flag

WM_CreateWindowAsChild(..., WM_CF_MOTION_X, ...);
```

You can set motion range for a window by assigning the value of the elements SnapX.

```
// horizontal motion range is LCD width

SnapX = 480;
```

5.2 RTC Setting

The NuMaker-Thermostat-N9H20 utilizes RTC to count time information.

```
// Declare a RTC, and set default time information.

RTC_Init();
```

Set the default time information and write to RTC.

```
RTC_TIME_DATA_T sCurTime;

sCurTime.u32Year = 2020;

sCurTime.u32cMonth = 4;

sCurTime.u32cDay = 10;

sCurTime.u32cHour = 6;
```

```
sCurTime.u32cMinute = 30;
sCurTime.u32cSecond = 50;
RTC_Write(RTC_CURRENT_TIME, &sCurTime);
```

5.3 PWM Backlight Control

The NuMaker-Thermostat-N9H20 utilizes PWM to control backlight.

```
// Declare a PWM and its frequency and level.
PWM_Open();

PWM_TIME_DATA_T sPt;

sPt.fFrequency = 1000;

/* High Pulse period : Total Pulse period = 1 : 100 */
sPt.u8HighPulseRatio = s_u8BLValue;

/* Set PWM Timer 0 Configuration */
PWM_SetTimerClk(PWM_TIMER0,&sPt);
```

5.4 Modbus Master Setting

The NuMaker-Thermostat-N9H20 utilizes high speed UART to control RS485. The Modbus Master RTU protocol is used to read Modbus Slave device to read temperature and on/off LED.

The RS485 needs Tx, Rx and RTS pin.

```
// Init GPIO PD.3 for nRTS control RS485
outp32(REG_GPDFUN, inp32(REG_GPDFUN) & ~0x00C0);
gpio_setportval(GPIO_PORTD, 0x8, 0x8);
gpio_setportpull(GPIO_PORTD, 0x8,0x8);
gpio_setportdir(GPIO_PORTD, 0x8, 0x8);
// For Uart-0 (GPD.1 --> TX, GPD.2 --> RX)
outp32(REG_GPDFUN, (inp32(REG_GPDFUN) & ~0x03C) | 0x14);
```

Set the baud rate, parity check, data bit and stop bit.

Note: Modbus Slave device needs to set the same values with Modbus Master.

```
WB_UART_T uart;

uart.uiBaudrate = 9600;

uart.uiParity = WB_PARITY_NONE;

uart.uiDataBits = WB_DATA_BITS_8;

uart.uiStopBits = WB_STOP_BITS_2;
```

5.5 Event Handler Control

The NuMaker-Thermostat-N9H20 utilizes emWin event handler for button operation.

The following events are sent from a BUTTON widget to its parent window:

Message	Description
WM_NOTIFICATION_CLICKED	BUTTON has been clicked.
WM_NOTIFICATION_RELEASED	BUTTON has been released.

Table 3 emWin BUTTON Widget Notification Codes

The symbols GUI_ID_BUTTON0 ~ GUI_ID_BUTTON9 define IDs that are used to make BUTTON widgets distinguishable from creation.

```
// Create a BUTTON widget

BUTTON_CreateUser(..., GUI_ID_BUTTON0, ...);


// Once BUTTON widget be clicked, WM_NOTIFICATION_CLICKED will be called

// Then you can check ID and do the next action

...

case WM_NOTIFICATION_CLICKED:

// Check which ID

if (Id == GUI_ID_BUTTON0+5)

{

// Do related action

}
```

5.6 BUTTON Widget Skin

You can change the appearance of widget. The method allows changing the look by using a dedicated bitmap which defines how the widgets are rendered.

The default emWin BUTTON widget skin is as follows.

Figure 5-1 Default emWin BUTTON Widget Skin

After changing the BUTTON widget skin, the button is changed as follows.

Figure 5-2 User Replaced Default BUTTON Widget Appearance to a Bitmap

```
// Create a default BUTTON widget

BUTTON_CreateUser(...);

// Then, register a callback function _ButtonSkinMenu1 to change skin

BUTTON_SetSkin(..., _ButtonSkinMenu1);


// In callback function draw bitmap case

case WIDGET_ITEM_DRAW_BITMAP:

// To change skin

GUI_DrawBitmap(...);
```

6 FAQ

6.1 How to replace image?

Use the sample width and height image and convert to c array. For example, you have a new *menu3.bmp*. Then, convert it to *menu3.c* and replace the original *menu3.c* and re-compile the project.

Note: The image file name, width and height need to be the same as the Thermostat's image.

6.2 How to convert from image file to c array?

In section 4.2, Thermostat contains an image file and converted c array. You can use emWin tool *BmpCvtNuvoton.exe* to open the image file and save it as .c and its format is High color (565), red and blue swapped.

Note: Make sure to utilize the latest emWin version. (>= V5.48)

6.3 Why sliding effect looks so slow and laggy?

It is caused by enabling the Modbus function. Disable the function if no Modbus Slave devices are connected to thermostat.

7 REVISION HISTORY

Date	Revision	Description
2019.12.30	1.00	1. Initially release.
2019.12.31	1.01	1. Added Folder Structure. 2. Added Design Guide. 3. Added FAQ.
2020.02.18	1.02	1. Supported N9H20K3.
2020.04.14	1.03	1. Added Modbus Master.
2021.05.04	1.04	1. Modified document file name. 2. Updated Overview and Installation Environment.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.