Arm® 926-EJS

32-bit Microcontroller

N9H20
Meter HMI
User Manual

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

www.nuvoton.com

Table of Contents

1 OVERVIEW

Meter HMI for N9H20 is a GUI reference implementation.

This document utilizes Nuvoton N9H20 series geneal-purpose microprcessor N9H20K5 (32MB DDR) to implement Meter HMI with emWin GUI library. Nuvoton emWin GUI library supports hardware JPEG, BitBLT and OSD.

Figure 1-1 Meter HMI Main Menu

2 FEATURES

2.1 Meter HMI Features

- Support Nuvoton MPU N9H20K5
- Supports hardware JPEG decoder for baseline decoding
- Supports hardware BitBLT rotation and OSD overlapping
- Supports high quality and contrast LCD panel with resolution up to 480 x 272
- Supports SEGGER licensed emWin GUI library
- Supports many popular image formats, e. g., PNG, GIF, JPG and BMP
- Supports user defined image as icon source

Figure 2-1 Meter HMI Vehicle Speed

Figure 2-2 Meter HMI Intake Air Temperature

Figure 2-3 Meter HMI Fuel Volume

3 INSTALLATION AND ENVIRONMENT

3.1 Installing N9H20 Non-OS BSP

First, download the latest N9H20 Non-OS 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.

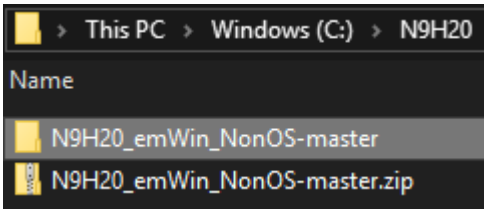

Figure 3-1 N9H20 Folder

The detailed information of N9H20 Non-OS BSP and emWin library can be found at "*N9H20_emWin_NonOS-master\N9H20 Readme.pdf*" and "*N9H20 emWin Quick Start Guide.pdf*" respectively.

3.2 Installing Meter HMI

First, download and unzip the latest "*N9H_emWin_Template-master.zip*" from https://github.com/OpenNuvoton/N9H_emWin_Template and copy "*Meter_N9H20_NonOS*" to the N9H20 sample path "*C:\N9H20\N9H20_emWin_NonOS-master\BSP\SampleCode\emWin*".

Then, open KEIL project file at "*C:\N9H20\N9H20_emWin_NonOS-master\BSP\SampleCode\emWin\Meter_N9H20_NonOS\KEIL\SimpleDemo.uvproj*" and start compiling. The executable binary is in "*C:\N9H20\N9H20_emWin_NonOS-master\BSP\SampleCode\emWin\Meter_N9H20_NonOS\Bin*", called "*conprog.bin*". Next, connect the USB cable between PC/NB and N9H20, then power on N9H20. Then copy "*conprog.bin*" to "*NAND1-1*" USB disk. Next, copy "*C:\N9H20\N9H20_emWin_NonOS-master\BSP\SampleCode\emWin\Meter_N9H20_NonOS\Bin\NAND1-2*.**" to "*NAND1-2*" USB disk. Finally, remove the USB disk safely and reboot N9H20.

3.3 System Requirements

- KEIL IDE V5.xx and above with professional license
- Nuvoton N9H20K5 480 x 272 demo board (NuDesign HMI-N9H20 + NuDesign TFT-LCD4.3)

4 FOLDER STRUCTURE

4.1 Meter HMI Folder Structure

The content of "*Meter_N9H20_NonOS*" is described as follows.

Folder	Description
Meter_N9H20_NonOS	Base folder ● Changelog.pdf is version history ● Meter_Reference_Implementation.pdf is user manual ● main.c is for Meter HMI main entry
Application	HMI folder ● NVT_Config.c is for Nuvoton platform ● NVT_Meter1.c is for Meter HMI image and data processing ● GUIConf.c is for emWin memory pool ● LCDConf.c is for emWin display driver control
Bin	Pre-built binaries folder ● conprog.bin is for Meter HMI execution file
Bin / NAND1-1	Resource folder ● conprog.bin is for Meter HMI execution file and identical with Bin's conprog.bin
Bin / NAND1-2	Dta resource folder ● Logo_128x64.dta is Nuvoton logo ● Meter1_480x272.dta is background of Meter HMI
Bin / 9To565	Any image convert to dta file format
Bin / LogGen	Any image convert to logo file format
Bin / Res	Image files
KEIL	Arm Keil MDK project folder

Table 4-1 Meter HMI Folder Structure

5 DESIGN GUIDE

Meter HMI 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

Note: the basic Meter HMI utilizes SEGGER's emWin GUI library. About SEGGER's emWin GUI library user manual can be found at "*C:\N9H20\N9H20_emWin_NonOS-master\BSP\ThirdParty\emWin\Doc\UM03001_emWin.pdf*".

5.1 Meter HMI Control

Figure 5-1 Meter HMI Control

Meter HMI control offers anti-aliasing and high resolution approach drawing lines and utilizes Trigonometric function lookup table calculating degrees and coordinates. It contains 1 meter background.

By "*NVT_Meter1.c*", you can assign user define dta image file to replace the original.

```
NVT_Meter1.c

//

// DTA image file full path (modify if needed)

//

// Logo

//

#define NVT_LOGO1 "D:\\Logo_128x64.dta"
```

```
//
// Meter HMI background
//
#define NVT_METER1 "D:\\Meter1_480x272.dta"
//
// Increase DTA buffer size if needed
//
static UINT8 s_u8Logo1ImageBuf[18 * 1024] __attribute__((aligned(32)));
static UINT8 s_u8Meter1ImageBuf[257 * 1024] __attribute__((aligned(32)));
```

5.2 Ping-Pong Buffer Control

Meter HMI utilizes ping-pong buffer control to avoid tearing and flickering. You need to declare two frame buffers:

```
main.c


UINT8 u8FrameBuf[XSIZE_PHYS*YSIZE_PHYS*2*2] __attribute__((aligned(32)));


UINT8 *u8FrameBufPtr;
UINT8 *u8FrameBuf2Ptr;
//
// Assign the first frame buffer to VPOST
//
u8FrameBufPtr  = (UINT8 *)((UINT32)u8FrameBuf | BIT31);
vpostLCMInit(&lcdFormat, (UINT32*)u8FrameBufPtr);
//
// Calculate the second frame buffer address
u8FrameBuf2Ptr = (UINT8 *)((UINT32)u8FrameBufPtr + 480 * 272 * 2);
//
// Meter HMI control loop
//
```

```
while (1)
{
.
.
.
//
// Before drawing, assign buffer to emWin first
//
if (s_i16PendingBuf == 0)
{
    LCD_SetVRAMAddrEx(0, (void *)u8FrameBuf2Ptr);
}
else
{
    LCD_SetVRAMAddrEx(0, (void *)u8FrameBufPtr);
}
//
// Start drawing
NVT_Meter1(0, 0);

NVT_CxtUpdate();
.
.
.
//
// Wait VSYNC
//
s_i16VsyncFlag = 0;
vpostEnableInt(eDRVVPOST_VINT);
while (s_i16VsyncFlag == 0);
```

```
t0 = GUI_X_GetTime() - t0;
//
// Calculate Frame-Per-Second
//
t1 = (U32)((1.0f / (float)t0) * 1000.0f);
}
```

5.3 Vehicle Speed, Intake Air Temperature and Fuel Volume Meter Control

Figure 5-2 Vehicle Speed Meter Control

List of all vehicle speed meter control definitions by the Meter HMI.

Note the vehicle speed range is from 0 to 240.

```
//
// High resolution setting
//
#define Circle_AA_Factor 6
//
// Degree Setting
//
#define Circle_StartDegreePoint -90
```

```
#define Circle_EndDegreePoint 360
//
// Radius Setting
//
#define BigCircle_rad 52
#define SmallCircle_rad 37
#define RPMSpeedCircle_rad 70
#define RPMSpeed_dregreeS -31
#define RPMSpeed_dregreeE 210
//
// X-Y position setting
//
#define VehicleSpeed_PositionX 168
#define VehicleSpeed_PositionY 138
#define Vehicle_Point_X0Y0  -(RPMSpeedCircle_rad-5)
#define Vehicle_Point_X1Y1 -(RPMSpeedCircle_rad+60)
```

List of all vehicle speed meter control APIs by the Meter HMI.

```
//
// Set vehicle speed
//
void Show_VehicleSpeed(volatile int degree_idex)
{
.
.
.
//
// Draw vehicle speed
//
Draw_VehicleLine(degree_idex);
```

```
}
```

Figure 5-3 Intake Air Temperature Meter Control

List of all intake air temperature meter control definitions by the Meter HMI.

Note the intake air temperature range is from -40 to 150.

```
//
// X-Y position setting
//
#define IntakeAirTemp_PositionX 370
#define IntakeAirTemp_PositionY 70
#define IntakeAirTemp_Point_X0Y0  -(BigCircle_rad-10)
#define IntakeAirTemp_Point_X1Y1 -(BigCircle_rad-15)
```

List of all intake air temperatue meter control APIs by the Meter HMI.

```
//
// Set intake air temperature
//
void Show_IntakeAirTemp_Position(volatile int degree_idex)
{
.
.
.
//
// Draw intake air temperature
//
Draw_IntakeAirRainbow(degree_idex);
```

```
}
```

Figure 5-4 Fuel Volume Meter Control

List of all fuel volume meter control definitions by the Meter HMI.

Note the fuel volume range is from 0 to 240.

```
//
// X-Y position setting
//
#define FuelVolume_PositionX 370
#define FuelVolume_PositionY 200
#define FuelVolume_Point_X0Y0  0
#define FuelVolume_Point_X1Y1 -40
```

List of all fuel volume meter control APIs by the Meter HMI.

```
//
// Set fuel volume
//
void Show_FuelVolume(volatile int degree_idex)
{
.
.
.
//
// Draw fuel volume
//
```

```
Draw_FuelVolumeLine(degree_idex);
```

6 FAQ

6.1 How to replace dta?

Use the same image filename, width and height and utilize 9To565 to convert to dta then copy to NAND1-2.

7 REVISION HISTORY

Date	Revision	Description
2020.12.14	1.00	1. Initially release.

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.