Arm® 926-EJS

32-bit Microcontroller

N9H30
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 N9H30 is a GUI reference implementation.

This document utilizes Nuvoton N9H30 series geneal-purpose microprcessor N9H30F61IEC to implement Meter HMI with emWin GUI library. Nuvoton emWin GUI library supports hardware JPEG, BitBLT and OSD.

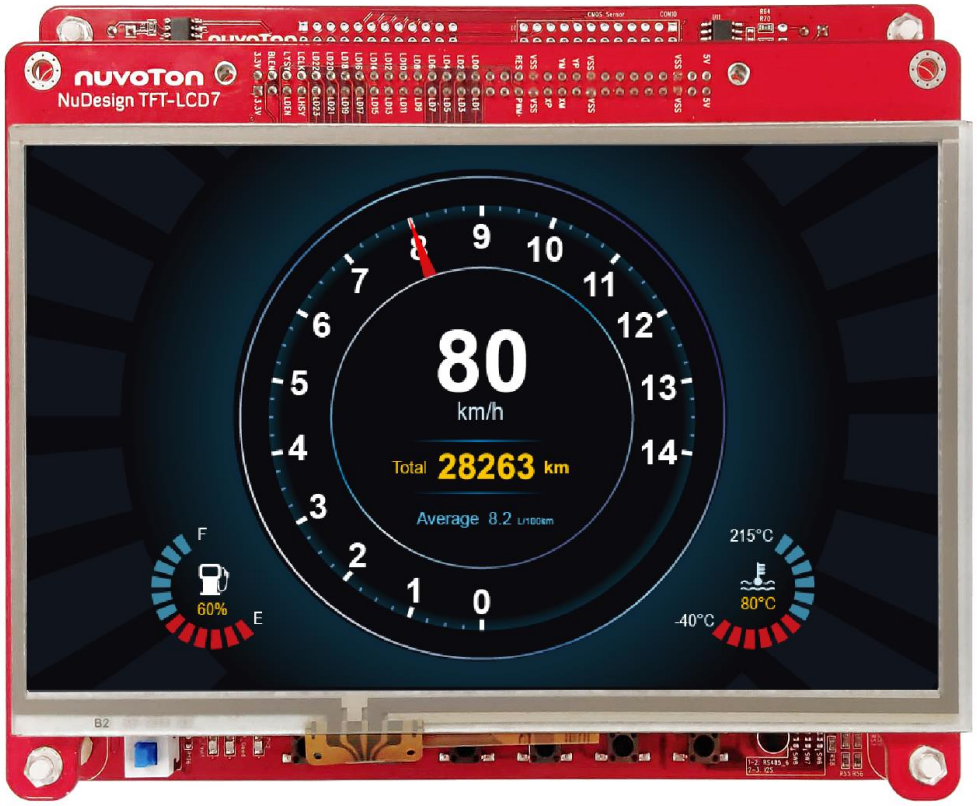

Figure 1-1 Meter HMI Main Menu

2 FEATURES

2.1 Meter HMI Features

- Support Nuvoton MPU N9H30
- 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 800 x 480
- 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

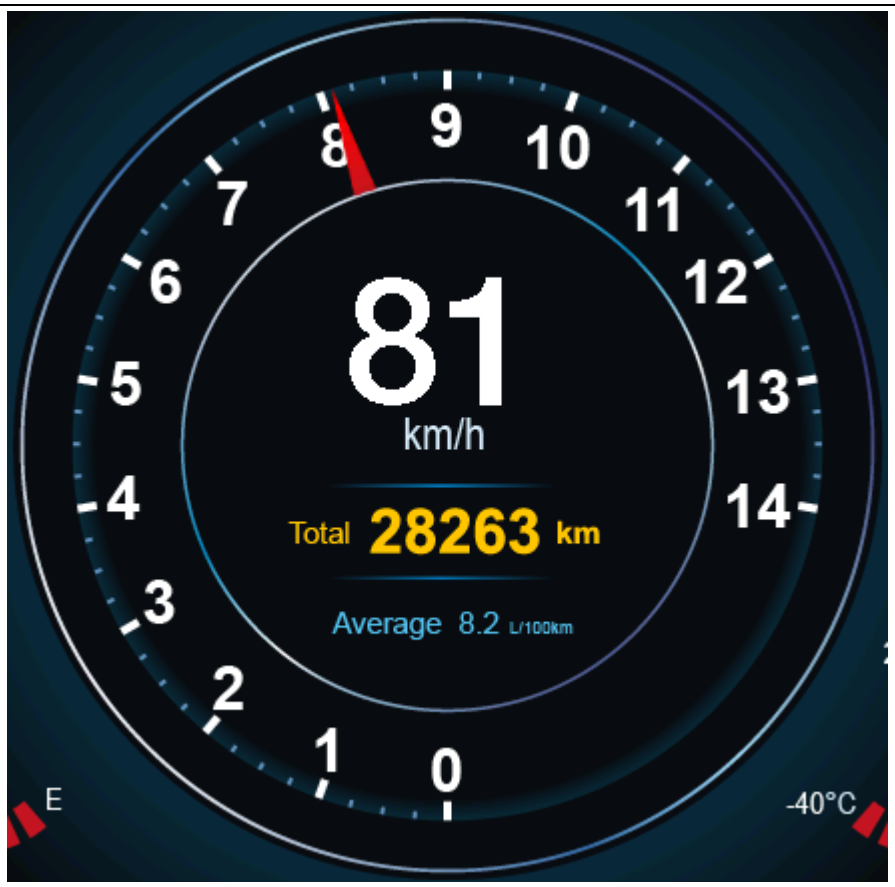

Figure 2-1 Meter HMI Vehicle Speed

3 INSTALLATION AND ENVIRONMENT

3.1 Installing N9H30 Non-OS BSP

First, download the latest N9H30 Non-OS BSP from https://github.com/OpenNuvoton/N9H30_emWin_NonOS and unzip "*N9H30_emWin_NonOS-master.zip*" to a working folder, e. g., unzip it to the path "*C:\N9H30*", where "*N9H30*" is the working folder.

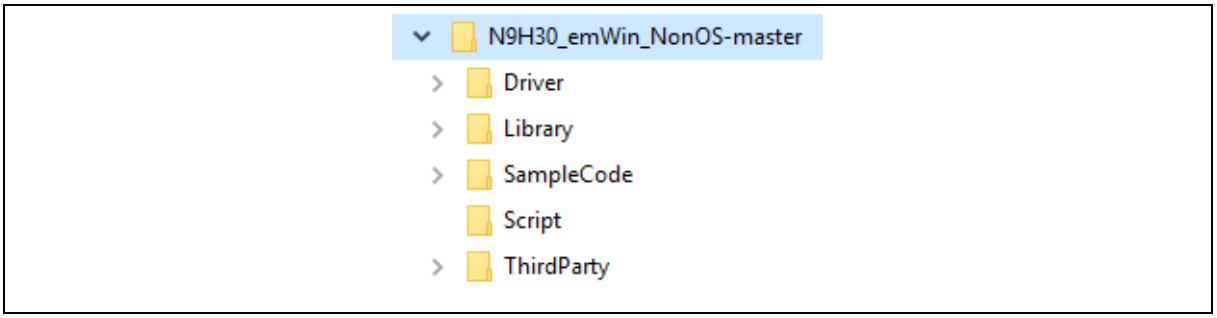

Figure 3-1 N9H30 Folder

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_N9H30_NonOS*" to the N9H30 sample path "*C:\N9H30\N9H30_emWin_NonOS-master\BSP\SampleCode*".

Then, open KEIL project file at "*C:\N9H30\N9H30_emWin_NonOS-master\BSP\SampleCode\Meter_N9H30_NonOS\KEIL\Meter_N9H30_NonOS.uvproj*", select the build option "Meter_N9H30_NonOS" and start compiling. The executable binary file is in "*C:\N9H30\N9H30_emWin_NonOS-master\BSP\SampleCode\Meter_N9H30_NonOS\Bin*", called "Meter_N9H30_NonOS.bin". Next, set USB booting of N9H30 board, connect the USB cable between PC/NB and N9H30, then power on N9H30. Run Windows tool NuWriter to burn the binary code into DDR/SRAM of board to test as follows.

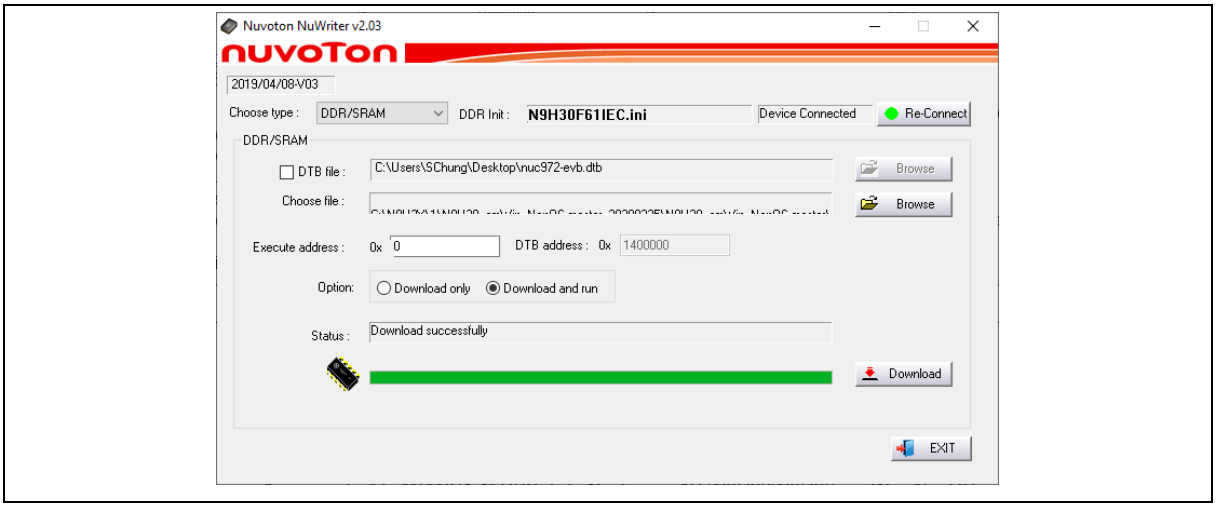

Figure 3-2 NuWriter

As to burn the code into NAND, please refer to the document "Nuvoton N9H30 Platform_Burn_Code.pdf " or "NUC970 N9H30 NuWriter User Manual.pdf" for more details.

3.3 System Requirements

- KEIL IDE V5.xx and above with professional license
- Nuvoton N9H30 800 x 480 demo board (NuDesign HMI-N9H30 + NuDesign TFT-LCD7)

4 FOLDER STRUCTURE

4.1 Meter HMI Folder Structure

The content of "*Meter_N9H30_NonOS*" is described as follows.

Folder	Description
Meter_N9H30_NonOS	Base folder ● Meter_Reference_Implementation.pdf is user manual ● main.c is for Meter HMI main entry
Application	HMI folder ● GUIDemo.c is for Meter HMI image and data processing ● Images.c is the C source code including all bitmap files ● LCDConf2.c is for emWin display driver control
Bin	Post built binaries folder ● Meter_N9H30_NonOS.bin is for Meter HMI execution file
Photo	Resource folder ● All bitmap files for Meter HMI
KEIL	Arm Keil MDK project folder
ThirdParty\emWin\Config	● GUIConf.c is for emWin memory pool

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 N9H30 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:\N9H30\N9H30_emWin_NonOS-master\ThirdParty\emWin\Doc\UM03001_emWin.pdf*".

5.1 Meter HMI

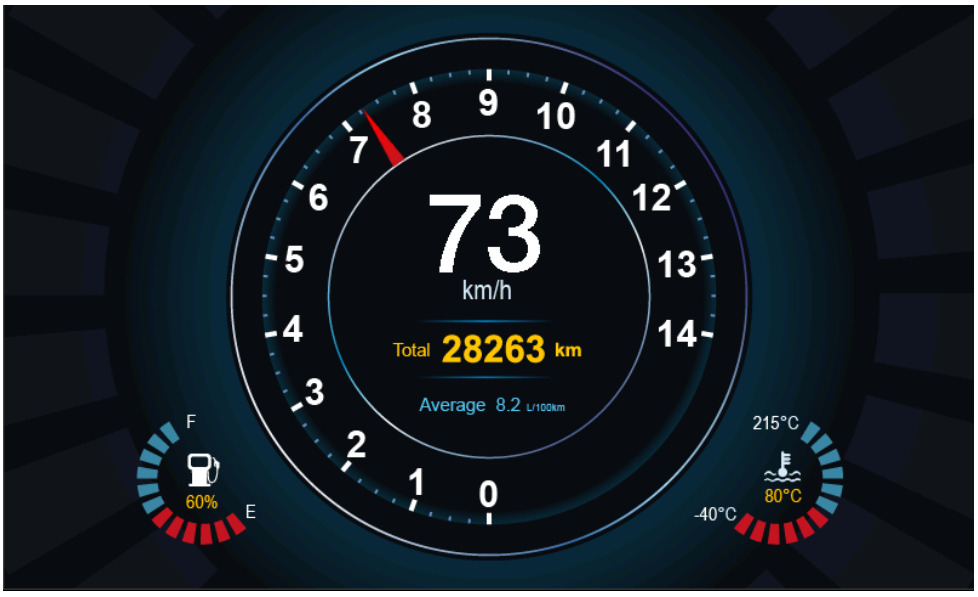

Figure 5-1 Meter HMI Control

Meter HMI offers anti-aliasing and high resolution approach drawing text and bitmap operations. Using Memory Device API to process degree rotation of bitmap needle, two memory devices operation and LCD display.

The background image of 800x480 is the file "Cluster_back_4.png" shown as follows

Figure 5-2 Background image of Meter HMI

If we update the changed image by using the 800x480 background image, the performance is very poor. So we use the trick to promote as follows.

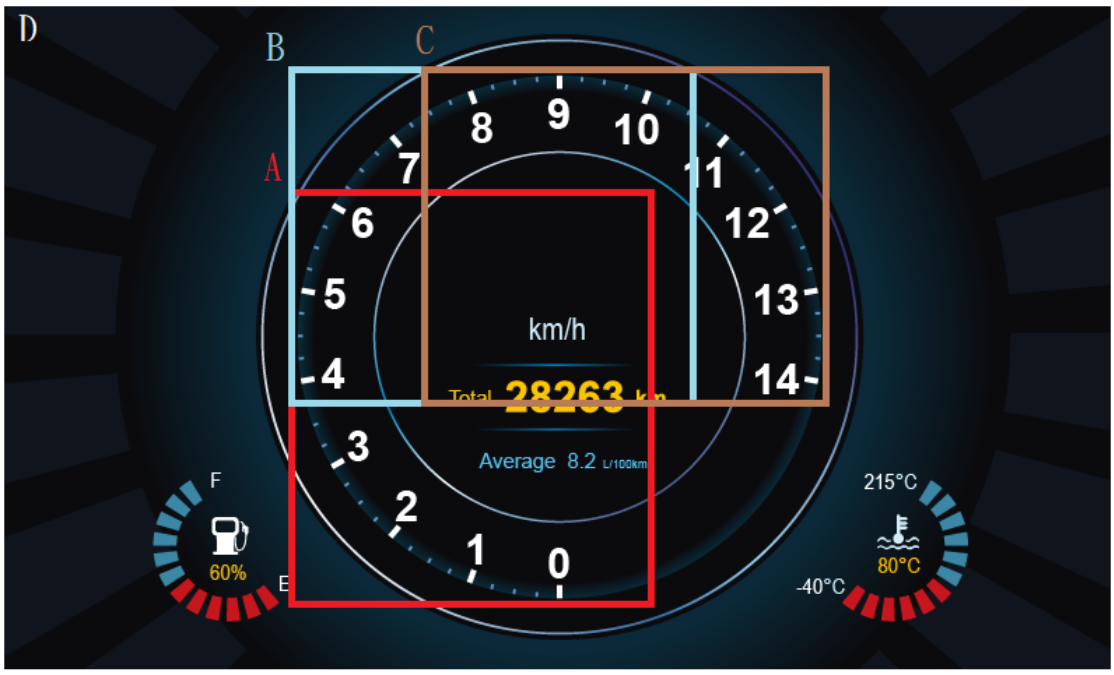

Figure 5-3 Four Rectangles

In file GUIDemo.c, we create four memory devices for four rectangles, the representation is the coordinates (left, top) and width X height, So rectangle D is (0, 0) and 800 X 480. Rectangle A is for the file FIG4.png, the representation is (208, 136) and 258 X 296, Rectangle B is for file FIG5.png, the representation is (208, 48) and 288 X 240, Rectangle C is for the file FIG6.png, and the represetation is (304, 408) and 288 X 240, we use tool BmpCvtNuvoton to exports C files for PNG files. The following code create four memory devices to save the four bitmap files for rectangles, four memory devices for the result of rotation.

```
    hScale    = GUI_MEMDEV_CreateFixed32(0, 0, bmCluster_back_4.XSize,
bmCluster_back_4.YSize);

    hScaleRot = GUI_MEMDEV_CreateFixed(0, 0, bmCluster_back_4.XSize,
bmCluster_back_4.YSize, GUI_MEMDEV_NOTRANS, GUI_MEMDEV_APILIST_32,
GUICC_M8888I);

    hNeedle   = GUI_MEMDEV_CreateFixed32(0, 0, _bmNeedle.XSize,
_bmNeedle.YSize);

      hFIG4  = GUI_MEMDEV_CreateFixed32(0, 0, 258, 296);

      hFIG4Rot = GUI_MEMDEV_CreateFixed(0, 0, 258, 296,
GUI_MEMDEV_NOTRANS, GUI_MEMDEV_APILIST_32, GUICC_M8888I);

      hFIG5  = GUI_MEMDEV_CreateFixed32(0, 0, 288, 240);

      hFIG5Rot = GUI_MEMDEV_CreateFixed(0, 0, 288, 240,
GUI_MEMDEV_NOTRANS, GUI_MEMDEV_APILIST_32, GUICC_M8888I);

      hFIG6  = GUI_MEMDEV_CreateFixed32(0, 0, 288, 240);

      hFIG6Rot = GUI_MEMDEV_CreateFixed(0, 0, 288, 240,
GUI_MEMDEV_NOTRANS, GUI_MEMDEV_APILIST_32, GUICC_M8888I);
```

5.2 The Rotation of Needle

The coordinates of needle is (400, 240), the center of rectangle D. We use tool BmpCvtNuvoton to open the file needle.png, counterclockwise rotation 90 degrees and export the C file as follows.

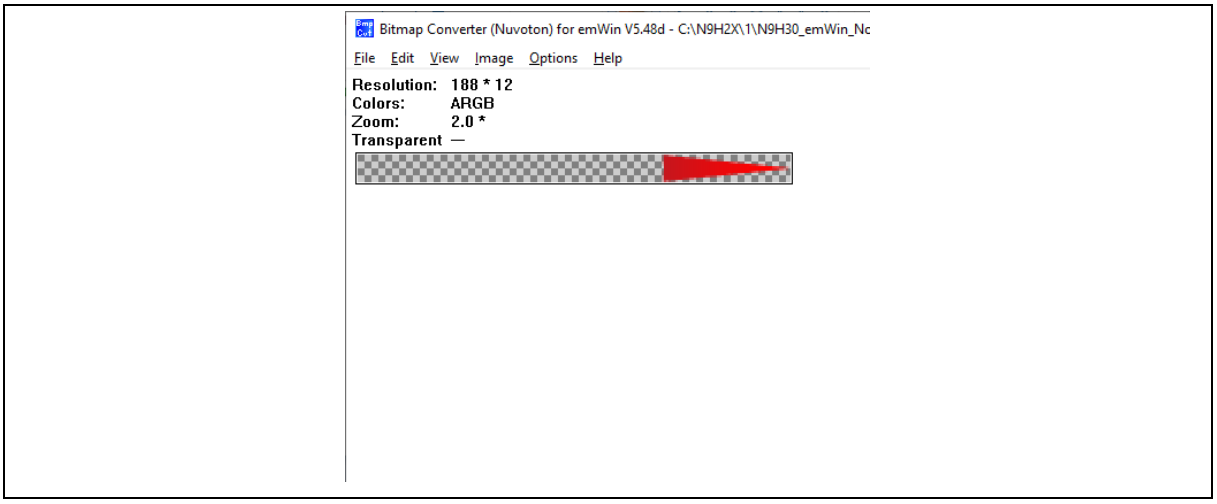

Figure 5-4 The needle of counterclockwise 90 degrees

The needle is located on X-axis of plane coordinates in mathematics, is also the location of 0 degree. The coordinates of circle center for the meter is (400, 240), the following code that needle rotates within the rectangle A.

```
static void _DrawFIG4Needle( int Angle, GUI_MEMDEV_Handle hDst) {
  int xSizeNeedle, ySizeNeedle, xSizeDst, ySizeDst;
  I32 SinHQ, CosHQ;
  //
  // Draw needle
  //
  xSizeNeedle = GUI_MEMDEV_GetXSize(hNeedle);
  ySizeNeedle = GUI_MEMDEV_GetYSize(hNeedle);
  xSizeDst =  384;     //440-28*2;
  ySizeDst =  208;          // 208;


  SinHQ = GUI__SinHQ((I32)((Angle) * (-1000)) + 90000);
  CosHQ = GUI__CosHQ((I32)((Angle) * (-1000)) + 90000);
  GUI_MEMDEV_RotateHQHR(hNeedle, hDst,
                        ((xSizeDst - xSizeNeedle) * 4) - (((xSizeNeedle)
* CosHQ) >> 14),
                        ((ySizeDst - ySizeNeedle) * 4) + (((xSizeNeedle)
* SinHQ) >> 14
                        (int)((Angle) * (-1000) + 270000), 1000);
}
.
```

How to obtain the values of variable “xSIzeDst”, and “ySizeDst” ? The coordinates (left, top) for rectangle A is (208, 136), so xSizeDst = (400 – 208) x 2 = 384, ySizeDst = (240 – 136) X 2 = 208. So the formula is xSizeDst = (X_center – left) X 2, ySizeDst = (Y_center – top) X 2. We could use the method for the needle rotation of other rectangles.

5.3 Meter Operation

The vehicle speed range is from 0 to 140. In the beginning set high resolution anti aliasing by the following code

```
      // Initialize high resolution anti aliasing
      GUI_AA_EnableHiRes();
```

```
      GUI_AA_SetFactor(6);
```

Memory device “hScale” is for rectangle D, memory device “hScaleRot” is rectangle D with needle rotation. Memory device “hFIG4” is for rectangle A, memory device “hFIG4Rot” is for rectangle A with needle rotation. Memory device “hFIG5” is for rectangle B, memory device “hFIG5Rot “ is for rectangle B with needle rotation. Memory device “hFIG6” is for rectangle C, memory device hFIG6Rot” is for rectangle C with needle rotation. Memory device “hNeedle” is for needle.

we put 800 x 480 image that needle rotates 0 degree by the following code

```
 speed = 0;
 angle = speed *2;
 GUI_MEMDEV_Select(hScaleRot);
 GUI_Clear();
 GUI_MEMDEV_Write(hScale);
 _DrawSpeed(speed);
  _DrawNeedle(angle, hScaleRot);
  GUI_MEMDEV_CopyToLCD(hScaleRot);
  GUI_MEMDEV_Select(0);
```

We set the following constant

```
#define T_MAX          6000   // 6 sec
#define MAX_SPEED      140
```

We could accelerate and brake the velocity by the following code

```
t0 = GUI_GetTime();
for (; (tDiff = GUI_GetTime() - t0) < T_MAX;) {
   tStart = GUI_GetTime();
   //
   // Calculate speed dependent on time
   //
   if (tDiff < (T_MAX >> 1)) {
      f = 1 - (float)tDiff / (T_MAX >> 1);
     f = f* f;
      speed = MAX_SPEED - (MAX_SPEED * f);
   } else {
      f = 1 - ((float)tDiff - (T_MAX >> 1)) / (T_MAX >> 1);
```

```
      f = f*f;
      speed = (MAX_SPEED * f);
    }
    angle = speed*2;
```

The velocity has three conditional expression, the first is 0 <= velocity < 50 for rectangle A, the second is 50 <= velocity < 90 for rectangle B and the third is 90 <= velocity <= 140 for rectangle C. The following code is the operation of the first conditional expression for rectangle A.

```
if ((speed >= 0)  && (speed < 50)) {
     GUI_MEMDEV_Select(hFIG4Rot);
    GUI_MEMDEV_WriteAt(hFIG4, 0, 0);
    _DrawFIG4Needle(angle,hFIG4Rot);
    _DrawFIG4Speed(speed);
     GUI_MEMDEV_CopyToLCDAt(hFIG4Rot,208, 136);
}
```

We could run the needle rotation smoothly by the following code.

```
#define T_MIN_FRAME  35


tUsed = GUI_GetTime() - tStart;
if (tUsed < T_MIN_FRAME) {
    GUI_X_Delay(T_MIN_FRAME - tUsed);
}
```

In the file main.c, we use the function LCD_Initial to set RGB565 for LCD as follows.

```
#ifdef _PANEL_FW070TFT_24BPP_
//    vpostSetVASrc(VA_SRC_RGB888);
    vpostSetVASrc(VA_SRC_RGB565);
#endif
```

It is better to set RGB565 instead of RGB888 for the performance.

It sets at least 7 MB memory for emWin GUI within the file GUIConf.c, otherwise the function fails to run.

```
#define GUI_NUMBYTES  (1024 * 1024 * 7)
```

6 REVISION HISTORY

Date	Revision	Description
2022.1.28	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.