Linux-based Digital Frame System

NuMicro® 32位系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例以電子相框為基礎產品，並在 Linux 平台上實現整體應用設計。系統內建簡易的 Web Server，使用者可透過瀏覽器上傳或更新照片至裝置中，形成一套直覺且方便的圖片更新流程，提供產品在操作端的維護與應用方式。
BSP 版本	Linux-5.10.x
開發平台	NuMaker-HMI-N9H30

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

1. 概述

本範例程式針對海外市場需求，將示範 Linux 環境下，快速完成顯屏應用，搭配 NuMaker-HMI-N9H30 開發平台，以電子相框為題，逐步完成介面設計與功能整合。善用作業系統優勢，展示 Web Server 應用，用戶可以透過網頁來更新或上傳照片，即使在手機或平板依舊能上傳照片。

此範例採新唐 GUI 開發平台 NuMaker-HMI-N9H30 開發板，其帶有尺寸為 7 吋的 TFT-LCD 電阻式觸控面板，解析度為 800x480。在開始前，需先架設 NuMaker-HMI-N9H30 Linux 開發環境，請參考 ARM9 系列的 NUC9x0 Linux Environment on VMware User Manual 文件進行開發前配置。

2. 程式介紹

本電子相框系統框架如下所示，可分成兩個部分。

Part 1 — 顯示功能

核心顯示功能，由主程式負責影像輪播，並從 photo 資料夾中依序讀取圖片檔案顯示於螢幕上；

Part 2 — 照片更新

新增 Web Server 負責更新影像，使用者可透過瀏覽器或手機端經由 Wi-Fi 上傳照片至 photo 資料夾，且在瀏覽器端會先執行圖片縮放與旋轉處理，使更新的影像符合顯示器解析度，無需在 Linux 端額外運算即可直接呈現，形成一套可即時更新播放的電子相框系統。

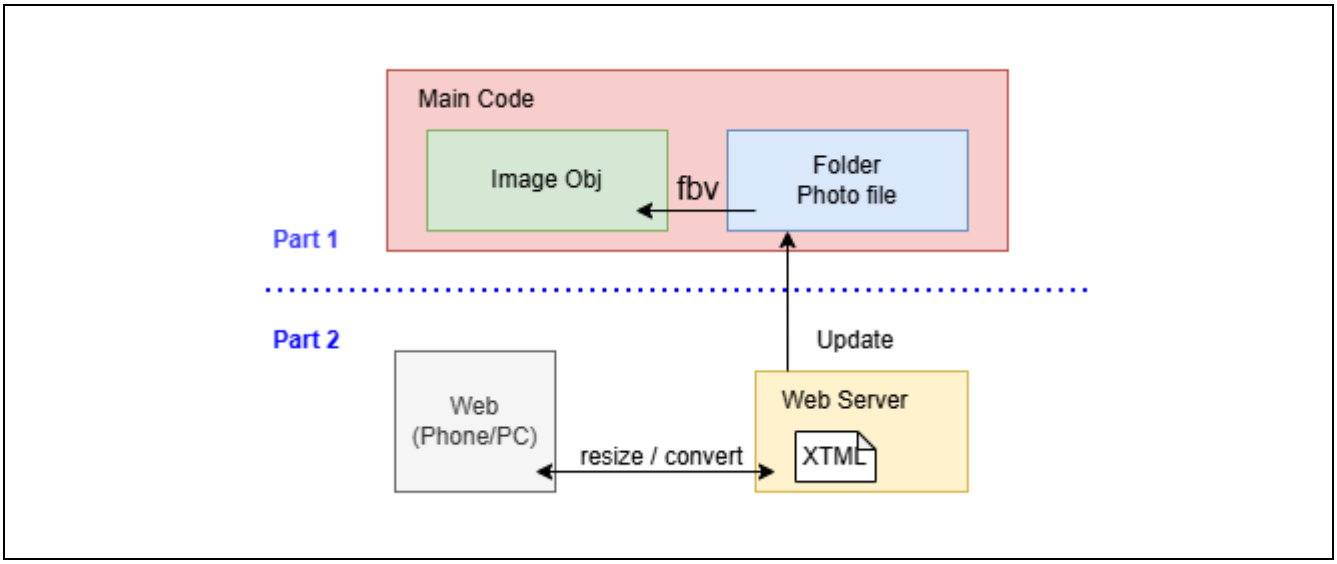

圖 2-1 電子相框系統框架

如何建立 Linux 中的 Rootfs，請參閱 *Linux 5.10 RootFS Build Guide*.。

本章節說明如何在 Linux/Buildroot 環境中實現「相片更新」功能。在 Buildroot 的 menuconfig 中啟用以下必要套件（可使用 “/ 鍵”搜尋套件名稱）：

- **BR2_PACKAGE_DHCPCD**

  用於取得 DHCP 指派的 IP 位址，負責網路介面的自動配置。
  Location :

  Target packages --- >

Networking applications --- >

[*] dhcpc

- **BR2_PACKAGE_NGINX**
  輕量級 Web 伺服器，負責提供相片上傳與管理所需的 HTTP 服務。

  Location:

  Target packages --- >

  Networking applications --- >

  [*] nginx

- **BR2_PACKAGE_FBV**

  簡易的 Framebuffer 圖片瀏覽工具，可直接在 Linux framebuffer 上顯示 JPG/PNG/BMP 圖片。

  Location:

  Target packages --- >

  Graphic libraries and applications (graphic/text) --- >

  [*] fbv

- **BR2_PACKAGE_LIGHTTPD**

  輕量化 Web Server，與 Nginx 類似但資源占用更低，適合嵌入式系統使用。可作為 HTTP 伺服器提供檔案上傳、靜態網頁、REST API 等功能。

  Location:

  Target packages --- >

  Networking applications --- >

  [*] lighttpd

Part 1: 主程式系統

主程式持續掃描指定資料夾*/opt/photo* 資料夾，搜尋副檔名為 *.png* 的圖片，並使用 **fbv** 指令在顯示器上依序顯示，形成輪播效果。程式亦會釋放 **scandir()** 所配置的記憶體，避免記憶體洩漏。此程式可用於電子相框或簡易顯示系統。

注意，程式中只判斷副檔名為 *.png* 的圖片並顯示，原因是在上傳上來的照片會限制儲存成.png 的格式，因此僅需要判斷*.png* 的圖片。

```
int main(void)
{
    struct dirent **namelist;
    int count;
    char fullpath[CMD_MAX_LEN];
    char cmd[CMD_MAX_LEN];

    while (1) {
        // Scan the photo directory and filter only *.png files.
        // 'filter_png' will return only files ending with .png
        // 'cmp_png' will sort them by filename or timestamp depending on your
implementation.
        count = scandir(PHOTO_DIR, &namelist, filter_png, cmp_png);
        if (count < 0) {
            perror("scandir failed");   // Print error if directory scan fails
            return -1;
        }

        // If no PNG files are found, wait 1 second and try again.
        if (count == 0) {
            printf("No PNG files found in %s\n", PHOTO_DIR);
            sleep(1);
            continue;
        }

        // Display each PNG file sequentially in sorted order.
        for (int i = 0; i < count; i++) {

            // Build the full file path: PHOTO_DIR + "/" + filename
            snprintf(fullpath, sizeof(fullpath), "%s/%s",
                     PHOTO_DIR, namelist[i]->d_name);

            // Use fbv (framebuffer viewer) to display the image.
            // Options:
            //   -a : automatically zoom to best fit
            //   -i : auto exit after display
            // Redirect all output to /dev/null to keep console clean.
            snprintf(cmd, sizeof(cmd), "fbv -a -i %s > /dev/null 2>&1", fullpath);
            system(cmd);   // Execute the command

            // Free each allocated dirent entry returned by scandir()
            free(namelist[i]);


            usleep(mseconds * 100);
        }


        sleep(3);

        // Free the namelist array allocated by scandir()
        free(namelist);
```

```
    }

    return 0;
}
```

Part 2: Web Server 處理流程

為 Web Server 處理程序，其中設置 *index.html* 來呈現使用者透過瀏覽器所呈現的 layout 布局，使用者可透過瀏覽器或手機端經由 Wi-Fi/Ethernet 上傳照片至 *photo* 資料夾，且在瀏覽器端會先執行圖片縮放與旋轉處理。

使用者的視窗佈局如下圖所示：

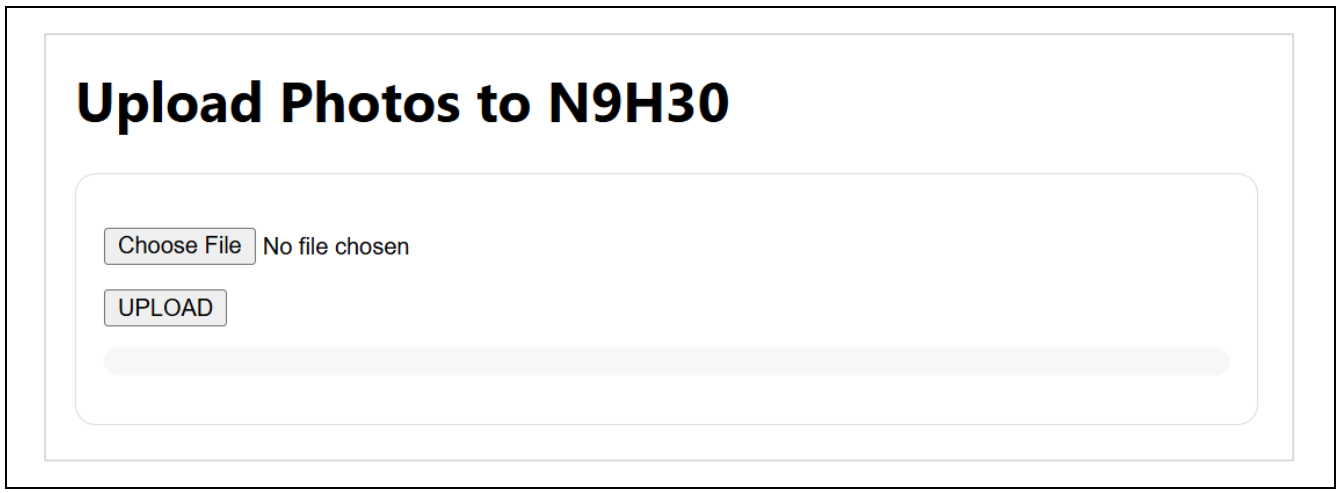

圖 2-2 網頁顯示頁面

網頁 **HTML** 檔案排版規劃如下：

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">

<!--
  Make layout responsive on mobile devices.
  Without this line, the UI will appear zoomed-in on smartphones.
-->
<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Photo Upload</title>

<style>
  /*
   * ===== Global Page Layout =====
   * Use system fonts for better performance.
   * Max width is set so content remains readable on large displays.
   */
```

```
  body {
    font-family: system-ui, -apple-system, Segoe UI, Roboto, Arial;
    max-width: 680px;           /* Limit UI width for better readability */
    margin: 40px auto;          /* Center vertically on desktop */
    padding: 0 16px;
  }

  /* Simple card-style container with soft rounded borders */
  .card {
    border: 1px solid #ddd;
    border-radius: 12px;
    padding: 16px;
  }

  /* Maintain consistent vertical spacing between each UI section */
  .row { margin: 12px 0; }

  /* Upload progress bar (hidden before upload starts) */
  progress {
    width: 100%;
    height: 16px;
  }

  /*
   * Status display area (upload result / error message).
   * `white-space: pre-wrap` keeps line breaks in log output.
   */
  #log {
    white-space: pre-wrap;
    background: #f7f7f7;
    padding: 8px;
    border-radius: 8px;
  }
</style>
</head>

<body>
<h1>Upload Photos to N9H30</h1>

<div class="card">
  <!-- File picker: only accept image formats -->
  <div class="row">
    <input id="file" type="file" accept="image/*">
  </div>

  <!-- Upload trigger button -->
  <div class="row">
    <button id="btn">UPLOAD</button>
  </div>

  <!-- Progress bar: hidden until upload begins -->
  <div class="row">
    <progress id="bar" max="100" value="0" hidden></progress>
  </div>

  <!-- Upload result / status message -->
```

```
  <div class="row" id="log"></div>
</div>
```

在網頁中的 **loadBitmap()**功能:：

loadBitmap() 函式負責從 **<input type="file">** 取得的圖片檔案中載入影像，並自動套用 **EXIF** 的旋轉資訊，以確保手機拍攝的照片不會出現上下顛倒或橫向顯示的問題。

```
/* ① Load image and apply EXIF orientation (if supported by browser) */
async function loadBitmap(file) {
  if ('createImageBitmap' in window) {
    try {
      return await createImageBitmap(file, { imageOrientation: 'from-image' });
    } catch (e) { /* Fallback if failed */ }
  }
  // Alternative method using Object URL
  const url = URL.createObjectURL(file);
  try {
    const img = new Image();
    img.decoding = 'async';
    img.src = url;
    await img.decode();
    return img;
  } finally {
    // URL will be revoked after drawing on canvas
  }
}
```

resizeToCanvas()函式：

resizeToCanvas()用途在於將載入的圖片依照螢幕解析度進行等比例縮放，並繪製至固定大小，同時保留原始比例以避免形變；在縮放後若仍有空白區域，會以指定的背景色填滿並將影像置中顯示，使輸出的影像完全符合 800×480 顯示尺寸，減少嵌入式系統的影像處理負擔。

```
/* ② Resize image to 800×480 (centered, keep aspect ratio, no distortion) */
function resizeToCanvas(bitmap, { maxW, maxH, background = '#000', upsize = true, mode =
'contain' }) {
  const iw = bitmap.width, ih = bitmap.height;
  let scale = mode === 'cover' ? Math.max(maxW/iw, maxH/ih) : Math.min(maxW/iw, maxH/ih);
  if (!upsize) scale = Math.min(scale, 1);
  const dw = Math.round(iw * scale);
  const dh = Math.round(ih * scale);

  // Create canvas buffer
  const canvas = document.createElement('canvas');
  canvas.width = maxW;
  canvas.height = maxH;

  const ctx = canvas.getContext('2d');
  ctx.fillStyle = background;
  ctx.fillRect(0, 0, maxW, maxH);
  ctx.imageSmoothingQuality = 'high';
```

```
  // Calculate offset (center image)
  const dx = Math.floor((maxW - dw) / 2);
  const dy = Math.floor((maxH - dh) / 2);
  ctx.drawImage(bitmap, dx, dy, dw, dh);

  return canvas;
}
```

canvasToBlob()函式:

canvasToBlob()負責將繪製完成的內容轉換成 Blob 格式的圖片檔，並限制輸出為 PNG 並調整壓縮品質；此方式可直接搭配 **XMLHttpRequest** 進行 HTTP PUT/POST 上傳到 N9H30，不需要再進行額外的轉檔或壓縮。

```
/* ③ Convert canvas to Blob (PNG or JPEG) */
function canvasToBlob(canvas, { type = 'image/png', quality = 0.85 } = {}) {
  // `quality` parameter is ignored by PNG but kept for compatibility
  return new Promise(resolve => canvas.toBlob(b => resolve(b), type, quality));
}
```

最後，程式設計完成後，需要將程式碼進行編譯。待編譯完成後，將生成執行檔案。該檔案可在 NuMaker-HMI-N9H30 開發平台上運行。以下說明 Linux 環境編譯流程。

Linux 環境中的交叉編譯流程:

1. 將 **EC_N9H30_Linux-based Digital Frame System_V1.00** 下的 **BuildCode** 移動到 N9H30 Linux 開發環境中**(Ubuntu)**。

2. 在進行交叉編譯時，需要先確認工具鏈（toolchain）是否設定正確。**Makefile** 中會使用 **CC** 指定的交叉編譯器來將 **SRCS** 裡的來源檔案編譯成 **TARGET** 的可執行檔，並可透過 **STRIP** 對產出的執行檔進行處理。

```
CC := arm-linux-gcc
STRIP := arm-linux-strip

TARGET = DigitalPhotoFrame
SRCS := DigitalPhotoFrame.c
```

3. 執行下列指令即可載入 Buildroot 所產生的交叉編譯器設定，讓之後的 **gcc**、**make** 等指令自動使用正確的 toolchain：

```
~/shared/N9H30/buildroot_2024$ source output/host/environment-setup
```

4. 移動到 **BuildCode** 資料夾，透過 **make** 指令進行編譯:

```
~/shared/N9H30/buildroot_2024/BuildCode$ make
```

若要清除編譯檔案或重新編譯，皆需執行以下指令:

```
~/shared/N9H30/buildroot_2024/BuildCode$ make clean
```

5. 編譯完成後，將生成執行檔 **DigitalPhotoFrame** 放置到*~output/target/opt* ，並重新 **pack**。該檔案可在 NuMaker-HMI-N9H30 開發平台上運行。

```
~/shared/N9H30/buildroot_2024$ make linux-rebuild && make
```

- N9H30 的系統配置說明:
- 在啟動 **nginx** 之前，系統需先取得網路 IP。透過 **udhcpc** 從 **eth0** 網路介面向 **DHCP** 伺服器請求並取得 IP 位址，之後才能啟動 Web Server。

```
#!/bin/sh

echo q | fbv -i /opt/logo_800x480.png
# 1. Check DHCP (try up to 5 times)
if udhcpc -i eth0 -n 5; then
    echo "DHCP OK – IP acquired."

    # 2. Create required folders for nginx (to avoid 'No such file or directory' errors)
    mkdir -p /var/log/nginx
    mkdir -p /var/cache/nginx/client-body
    mkdir -p /run/nginx

    # 3. Start nginx
    echo "Starting nginx..."
    nginx -t
    nginx &

    # 4. Show current IP address
    IP=$(ifconfig eth0 | awk '/inet /{print $2}')
    echo "Server running at: http://$IP/"

    # 5. Start the Digital Photo Frame application
    ./DigitalPhotoFrame &
else
    #  DHCP failed → show no-network warning image
    echo "DHCP FAILED – showing no network image..."
    fbv /opt/no_network.png
    exit 1
fi
```

- Nginx 配置說明:

```
http {
    include       mime.types;
    default_type  application/octet-stream;

    sendfile        on;
    keepalive_timeout  65;
```

```
    server {
        listen       80 default_server;
        server_name  _;

        # Web root for static files (HTML, JS, CSS, etc.)
        root  /opt/www;
        index index.html;

        # Home page: always serve index.html
        location = / {
            try_files /index.html =404;
        }

        # SPA-style routing:
        location / {
            try_files $uri $uri/ /index.html;
        }

        # === Upload folder: support HTTP PUT ================================
        # URL:  /upload/<filename>
        # Path: /opt/photo/<filename>
        location ^~ /photo/ {
        root /opt;

        client_max_body_size 10M;
        dav_methods PUT;
        create_full_put_path on;
        dav_access group:rw all:r;

        autoindex on;
        autoindex_localtime on;
    }
        # ====================================================================

        # Captive portal / OS connectivity checks -> redirect to main page
        location = /hotspot-detect.html       { return 302 /index.html; }
        location = /library/test/success.html { return 302 /index.html; }
        location = /generate_204              { return 302 /index.html; }
        location = /gen_204                   { return 302 /index.html; }
        location = /clients3/generate_204     { return 302 /index.html; }
        location = /connecttest.txt           { return 302 /index.html; }
        location = /ncsi.txt                  { return 302 /index.html; }

        # Simple error page
        error_page 500 502 503 504 /50x.html;
        location = /50x.html { }
    }
}
```

- 在首次設定或建立編譯環境時，系統於開機後會自動執行 **emWin AppWizard** 的範例程式。因此，若需改為啟動自訂應用程式，需先修改以下檔案後重新 **pack**：

```
~buildroot_2024/board/nuvoton/nuc9x0/rootfs-emwin/etc/init.d
```

- 新增啟動的程式名稱或腳本:

```
#!/bin/sh
export TSLIB_CONFFILE=/etc/ts.conf
export TSLIB_TSDEVICE=/dev/input/event0
export TSLIB_CALIBFILE=/etc/pointercal
export LD_LIBRARY_PATH=/lib/ts/
/sbin/mdev -s
cd opt
./init.sh  &
```

3. 軟體與硬體需求

3.1 軟體需求

- Embedded OS: Linux 5.10 版本

3.2 硬體需求

- 電路元件
  - NuMaker-HMI-N9H30

4. 目錄資訊

目錄結構如下圖所示。

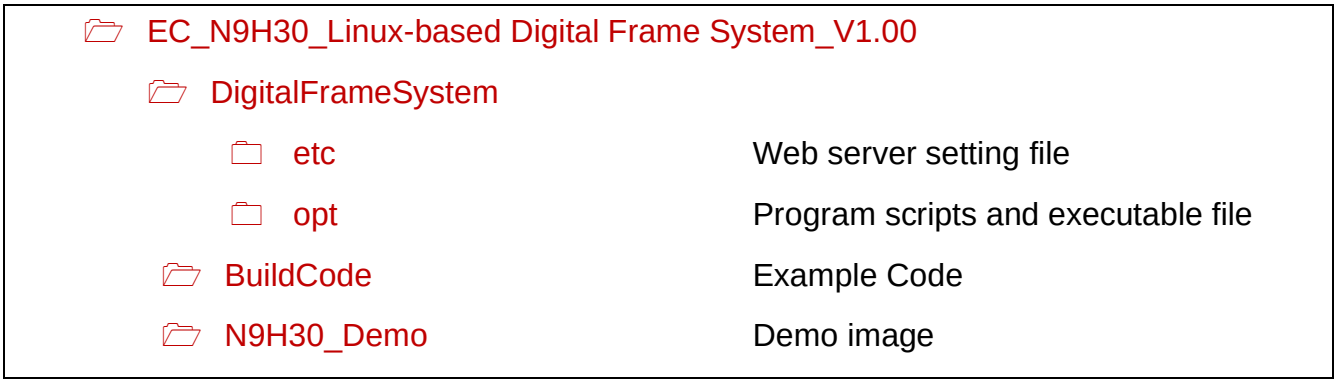

圖 4-1 目錄資訊

5. 範例程式執行

1. 透過 NuWriter 工具，將 *N9H30_Demo* 目錄中的 *n9h30image* 檔案燒錄到開發平台

   可以透過燒錄到 DDR 來測試本範例功能。如果是燒錄 *n9h30image*，本範例程式會在開機完成後自動執行。

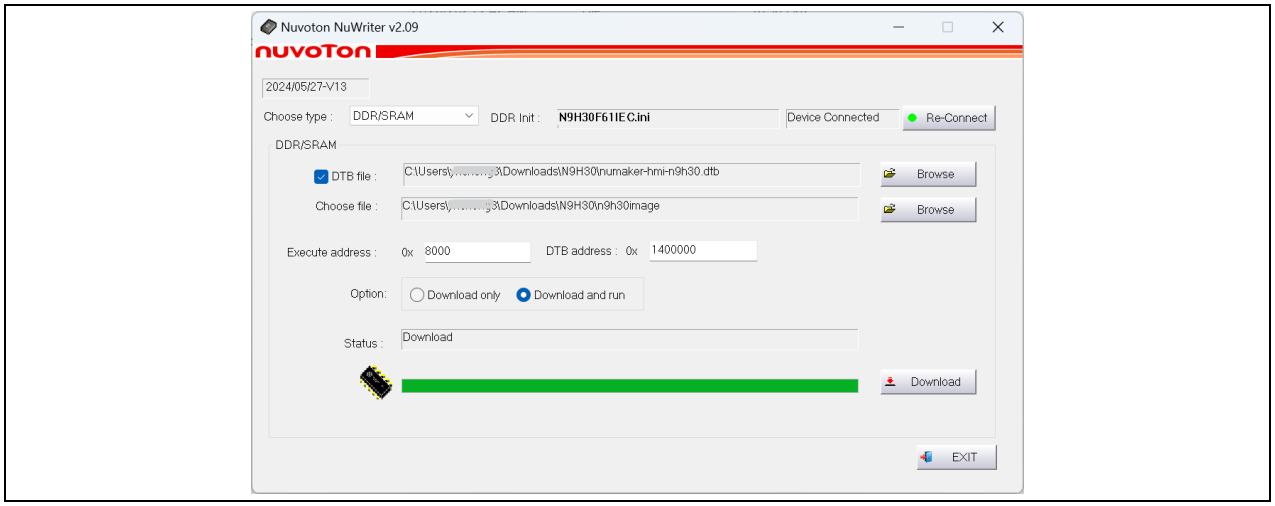

圖 5-1 NuWriter 燒錄工具

2. 在啟動 NuMaker-HMI-N9H30 前，需先連接乙太網路至 **eth0**。

3. 將手機與 NuMaker-HMI-N9H30 連接同一個網路，用戶可以上傳更新照片。

4. 用戶可以透過 **ifconfig** 查看當前 IP。

```
~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:00:01:92
          inet addr:192.168.0.32  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe00:192/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1912 errors:0 dropped:0 overruns:0 frame:0
          TX packets:290 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:32
          RX bytes:375753 (366.9 KiB)  TX bytes:26060 (25.4 KiB)
```

圖 5-2 IP 查詢

5. 透過瀏覽器開啟 192.168.0.32 網址。

圖 5-3 網頁更新畫面

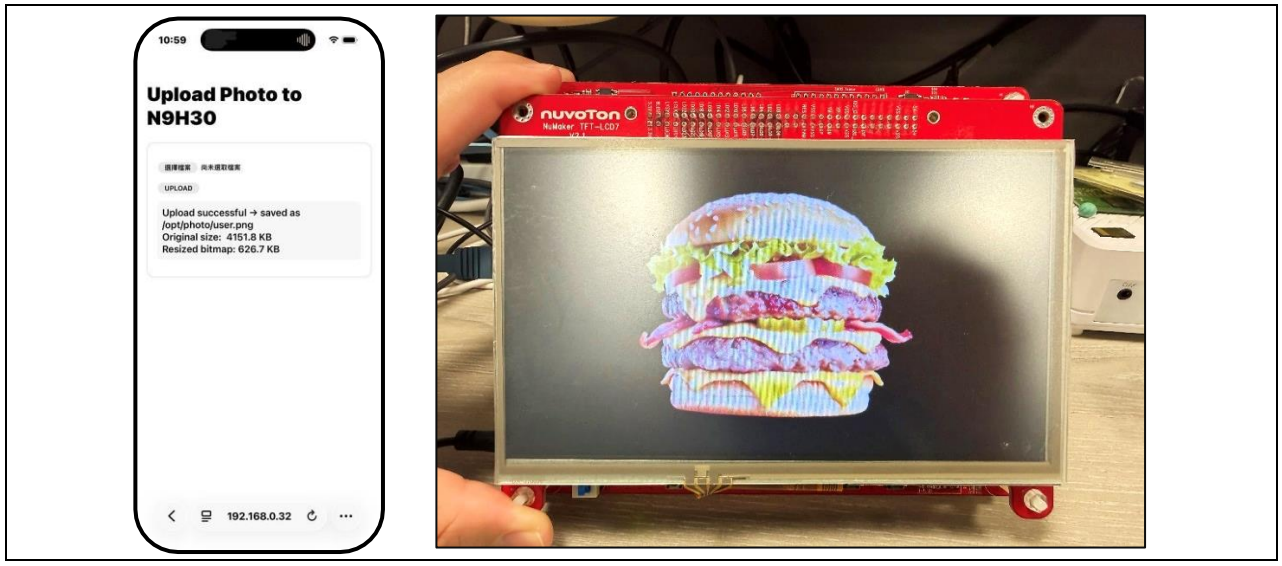

圖 5-4 Digital Frame System 執行結果及手機上傳更新頁面

用戶可依照下列步驟自行編譯與部署系統：

1. 開啟 N9H30 Linux 編譯環境，並將 **DigitalFrameSystem** 中的 *etc* 及 *opt* 資料夾複製到 *~output/target* 下的對應資料夾。重新 pack 映像檔後燒錄到 NuMaker-HMI-N9H30。此步驟流程可參考第二章節。

2. 進入 NuMaker-HMI-N9H30，輸入以下指令：

   移動到 *opt* 資料夾

   ```
   cd opt
   ```

   透過以下指令執行檔案：

   ```
   ./init.sh  &
   ```

6. 修訂紀錄

Date	Revision	Description
2025.09.10	1.00	初始發布。

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.