# Qt Overlay on Video

NuMicro® 64/32位元系列微控制器範例代碼介紹

## 文件資訊

| | |
|---|---|
| 應用簡述 | 本範例程式說明如何編寫 Qt 應用程式，此應用程式同時於螢幕底層撥放影片並於上層覆蓋互動元件，透過互動元件控制影片播放。 |
| BSP 版本 | Linux-5.10.x |
| 開發平臺 | NuMaker-HMI-MA35D1-S1 |

*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

混合顯示是計算機圖學中的一種技術，用於將多個圖像或圖層組合輸出成單一圖片，如圖 1-1 所示，通常應用於影片與圖像處理的應用程式，例如：影片合成。

本範例程式示範如何同時播放影片和運行 Qt 應用程式並顯示於螢幕上，影片和 Qt 應用程式分別輸出至不同的 Frame Buffer，如 Frame Buffer 0、Frame Buffer 1，並透過顯示器的驅動程式提供的混合渲染功能進行混合，再將混合後的圖顯示於螢幕上。MA35D1 顯示器驅動提供兩種混合渲染功能：Color Key 與 Alpha Blending，此範例程式則是使用 Alpha Blending。

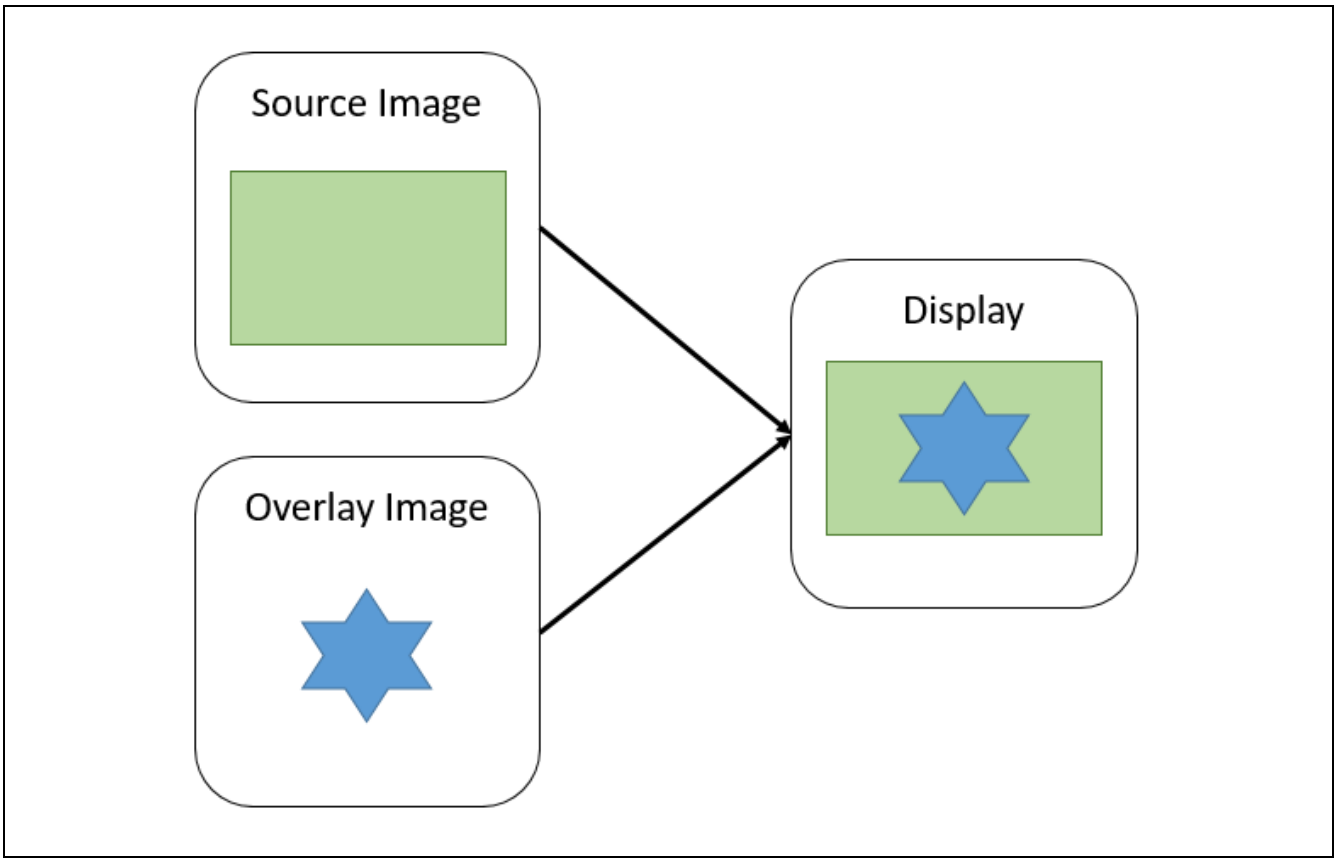


圖 1-1 圖像混合示意圖

## 1.1 軟體架構

軟體實現架構如圖 1-2 所示，Qt 應用程式使用 Gstreamer 函式庫播放影片，Gstreamer 則會使用 VC8000 硬體解碼器解碼影片並直接輸出至 Frame Buffer 0，Qt 使用 Qt 繪圖函式庫繪製需要顯示在影片上的元件輸出至 Frame Buffer 1，後續則交由顯示控制器混合影片與元件。

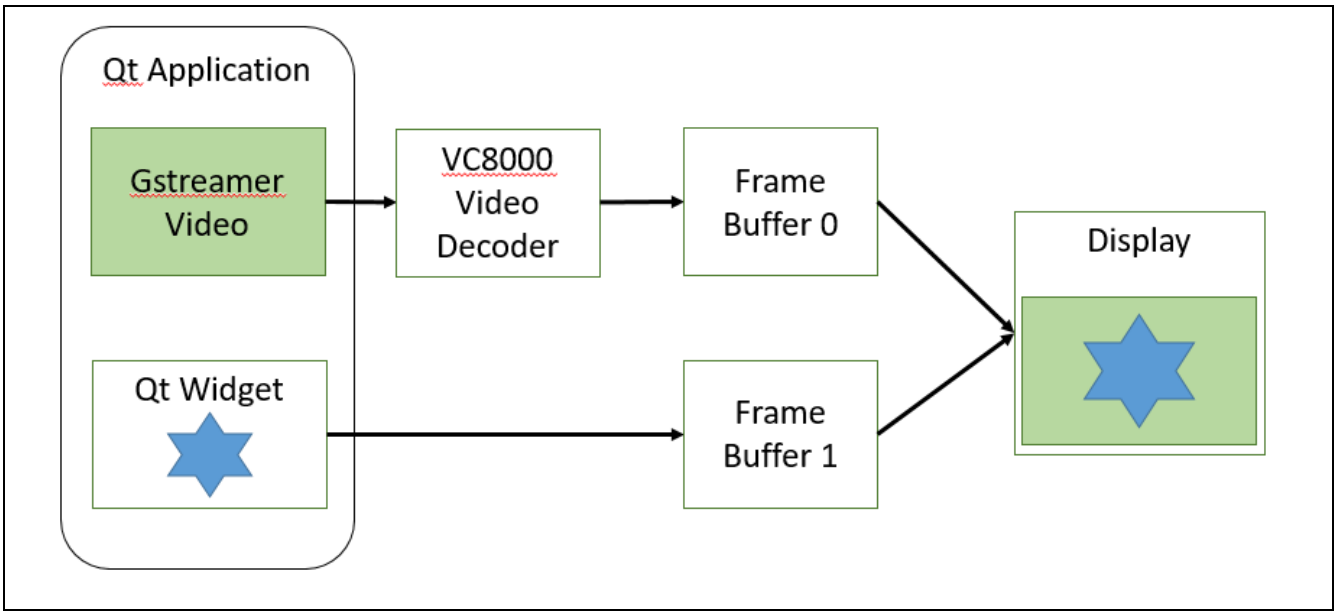


圖 1-2 軟體架構圖

## 1.2 驅動設定

### 1.2.1 顯示器設備樹設定

確認顯示器的解析度與開啟相關 overlay 選項，使用者可根據自身螢幕解析度調整顯示驅動解析度，以下使用解析度 1024*600 示範。

```
/* Ensure overlay related setting */
&display {
    ....
    overlay-en = <1>; /* Enable overlay function */
    overlay-pixel-fmt = <6>; /* Set RGB888 format */
    overlay-width = <1024>; /* Set overlay resolution width */
    overlay-height = <600>; /* Set overlay resolution height */
    overlay-rect-tlx = <0>; /* Set the starting coordinate of the x-axis */
    overlay-rect-tly = <0>; /* Set the starting coordinate of the y-axis */
    ....
};
```

## 2. 使用者程式介紹

此範例程式需使用 Qt Creator 來編輯與編譯 Qt 應用程式，根據以下步驟操作：

1. Gstreamer 與 Qt 是一個跨平台的多媒體框架開發平台，其在 MA35D1 上的安裝設定請參考 MA35D1 Buildroot Quick Start 的 Environment Setup 章節，Yocto 預設已經包含 Qt 與 Gstreamer，因此不用額外設定

2. 根據 MA35D1 HMI GUI Development Environment 裡的 Qt Creator Cross-Compile Environment Setup 章節來建立 Qt Creator 環境

3. 打開 Qt Creator，選擇 Open Project

4. 複製 EC_MA35D1_Qt_Overlay_on_Video_V1.00\Src\MA35D1_Qt_Overlay_on_Video 專案至 Linux 環境並開啟，如圖 2-1

5. 雙擊 MA35D1_Qt_Overlay_on_Video.pro 打開

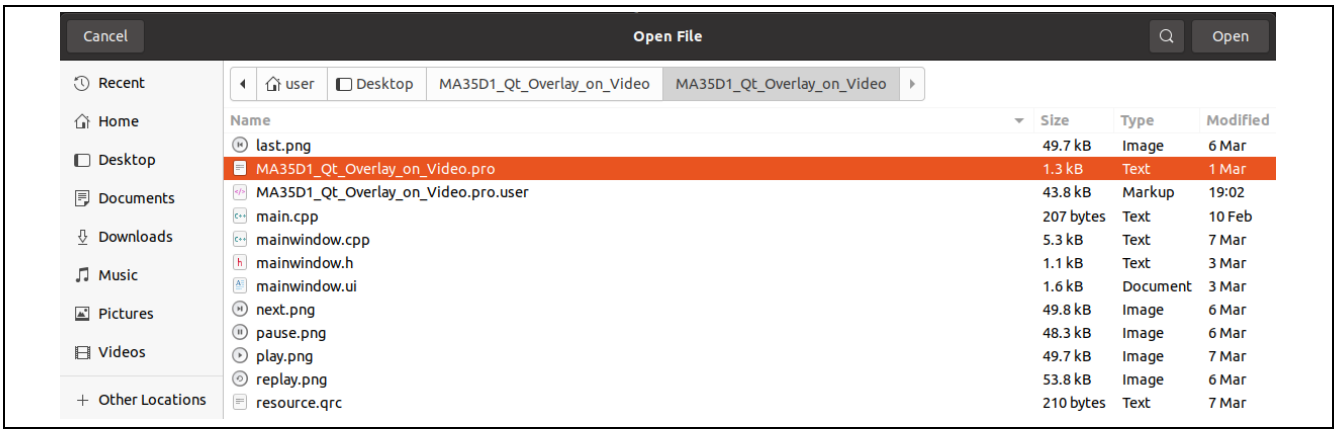


圖 2-1 Qt 專案示意圖

主要的 UI 控制行為都由 mainwindow.cpp 操控，此程式碼會將路徑/opt/videos 裡面的影片使用 Gstreamer 輪播，繪製 Qt 元件並輸出。主要控制行為分為：暫停、回復、重新播放、上一個與下一個，按下對應的按鈕即會觸發相對應的動作。

```
#include <QApplication>
#include <QDebug>
#include <QDir>
#include <stdio.h>
#include <stdlib.h>
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QTouchEvent>
```

```
#include <QThread>
#include <QTimer>
#include <gst/gst.h>

/* Define Videos location */
#define VIDEOS_FD_PATH "/opt/videos/"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    /* Gstreamer Setting */
    cur_video_idx = 0;
    QDir dir(QString::fromStdString(VIDEOS_FD_PATH));
    dir.setFilter(QDir::Files | QDir::Hidden | QDir::NoSymLinks);
    dir.setSorting(QDir::Size | QDir::Reversed);
    v_list = dir.entryInfoList();
    QString v_path = "gst-launch-1.0 filesrc location=" +
v_list.at(cur_video_idx).filePath() + " ! qtdemux name=demux demux.audio_0 ! queue !
decodebin ! audioconvert ! audioresample ! autoaudiosink demux.video_0 ! queue !
decodebin ! nufbdevsink ! fakesink sync=true ";
    gst_init (NULL, NULL);
    pipeline = gst_parse_launch (v_path.toStdString().c_str(), NULL);

    /* Set Icon */
    ui->play_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/pause.png); background-color: transparent;}");
    ui->replay_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/replay.png); background-color: transparent;background-position: center;}");
    ui->next_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/next.png); background-color: transparent;}");
    ui->last_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/last.png); background-color: transparent;}");

    /* This timer is used to auto replay the video */
    thread = new QThread(this);
    timer = new QTimer(0);
    timer->setInterval(1000);
    timer->moveToThread(thread);
    connect(timer, SIGNAL(timeout()), this, SLOT(video_update()),Qt::DirectConnection);
    connect(thread, SIGNAL(started()), timer,SLOT(start()));
    thread->start();

    /* Transparent Background */
    this->setAttribute(Qt::WA_TranslucentBackground, true);

    /* Start to Play Video */
    gst_element_set_state (pipeline, GST_STATE_PLAYING);
    playing = true;
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

```
void MainWindow::video_update()
{
    bus = gst_element_get_bus(pipeline);
    msg = gst_bus_timed_pop_filtered (bus,
GST_CLOCK_TIME_NONE,(GstMessageType)(GST_MESSAGE_ERROR | GST_MESSAGE_EOS |
GST_MESSAGE_UNKNOWN ));
    switch (GST_MESSAGE_TYPE (msg)) {
    case GST_MESSAGE_ERROR:
        qDebug() << "Stream Error";
        next_video();
        break;

    case GST_MESSAGE_EOS:
        qDebug() << "End-of-Stream";
        gst_element_set_state (pipeline, GST_STATE_NULL);
        gst_object_unref (pipeline);
        gst_message_unref (msg);
        gst_object_unref (bus);
        next_video();
        break;

    default:
        break;
    }

}

/* Play Video Action */

void MainWindow::resume_video()
{
    gst_element_set_state (pipeline, GST_STATE_PLAYING);
}

void MainWindow::pause_video()
{
    gst_element_set_state (pipeline, GST_STATE_PAUSED);
}

void MainWindow::replay_video()
{
    if(v_list.size() == 0)
        return;
    cur_video_idx--;
    cur_video_idx = cur_video_idx < 0 ? v_list.size()-1 : cur_video_idx;
    gst_element_post_message (pipeline, gst_message_new_eos(GST_OBJECT(pipeline)));
}

void MainWindow::next_video()
{
    if(v_list.size() == 0)
        return;
    cur_video_idx++;
    cur_video_idx = (cur_video_idx)%v_list.size();
    play_video();
```

```
}

void MainWindow::last_video()
{
    if(v_list.size() == 0)
        return;
    cur_video_idx-=2;
    if(cur_video_idx <= -2)
        cur_video_idx = v_list.size()-2;
    gst_element_post_message (pipeline, gst_message_new_eos(GST_OBJECT(pipeline)));
}

void MainWindow::play_video()
{
    gst_init (NULL, NULL);
    QString v_path = "gst-launch-1.0 filesrc location=" +
v_list.at(cur_video_idx).filePath() + " ! qtdemux name=demux demux.audio_0 ! queue !
decodebin ! audioconvert ! audioresample ! autoaudiosink demux.video_0 ! queue !
decodebin ! nufbdevsink ! fakesink sync=true ";
    pipeline = gst_parse_launch (v_path.toStdString().c_str(), NULL);
    gst_element_set_state (pipeline, GST_STATE_PLAYING);
}


void MainWindow::on_play_button_clicked()
{

    if(playing == true)
    {
        qDebug() << "Pause";
        pause_video();
        ui->play_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/play.png); background-color: transparent;}");
    }
    else
    {
        qDebug() << "Resume";
        resume_video();
        ui->play_button->setStyleSheet("QPushButton{ border-style: none; border-image:
url(:/pause.png); background-color: transparent;}");
    }
    playing = !playing;
}

void MainWindow::on_replay_button_clicked()
{
    qDebug() << "Replay";
    replay_video();
}

void MainWindow::on_next_button_clicked()
{
    qDebug() << "Next";
    gst_element_post_message (pipeline, gst_message_new_eos(GST_OBJECT(pipeline)));
}

void MainWindow::on_last_button_clicked()
```

```
{
    qDebug() << "Last";
    last_video();
}
```

圖 2-2 為 Qt 專案中的 UI 設計圖，統一將元件放置於螢幕下方，全部大小為 1024*600，每個按鈕大小為 100*100。

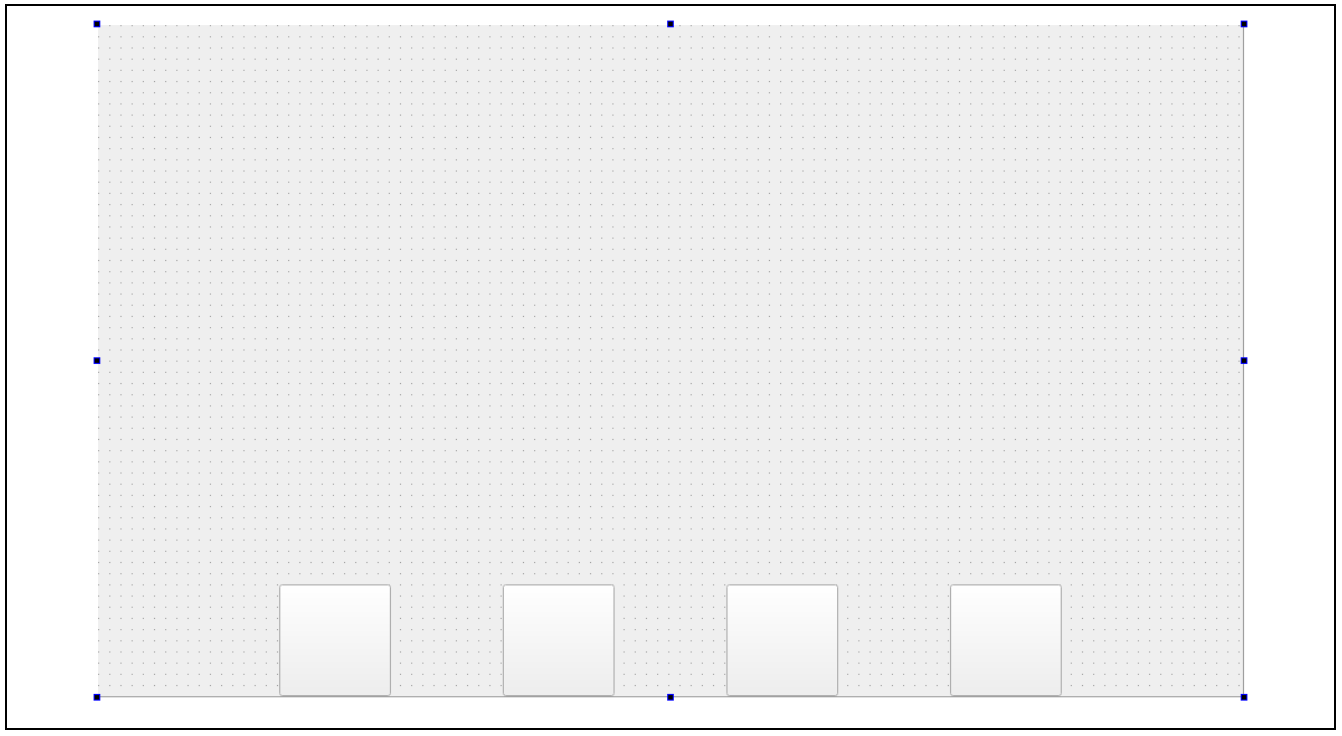

圖 2-2 mainwindow.ui 規劃圖

# 3. 軟體與硬體需求

## 3.1 軟體需求

- BSP 版本
  - Linux-5.10.x

## 3.2 硬體需求

- 電路元件
  - NuMaker-HMI-MA35D1-S1

# 4. 目錄資訊

| | |
|---|---|
| EC_MA35D1_Qt_Overlay_on_Video_V1.00 | |
| Src | Source files of Qt project |
| DT-Binding | Device tree requirements on the contents of driver |

圖 4-1 目錄資訊

# 5. 範例程式執行

1. 設定顯示器設備樹(請參照 1.2.1 顯示器設備樹設定)
2. 重新編譯 MA35D1 Image 並燒錄
3. MA35D1 開機後，參考 MA35D1 HMI GUI Development Environment 來設置 Qt 與 tslib 環境變數
4. 複製 MA35D1_Qt_Overlay_on_Video 應用程式到開發板
5. 複製影片至 /opt/videos/
6. 執行以下命令

```
./MA35D1_Qt_Overlay_on_Video -platform linuxfb:fb=/dev/fb1
```

7. 執行後畫面如圖 5-1

圖 5-1 MA35D1_Qt_Overlay_on_Video 執行畫面

# 6. 修訂紀錄

| Date | Revision | Description |
|---|---|---|
| 2024.02.16 | 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.*