ĐOÀN THANH NIÊN CỘNG SẢN HỒ CHÍ MINH

**BAN CHẤP HÀNH TP. HỒ CHÍ MINH**

**——🙠🙢🟏🙠🙢 ——**

**CÔNG TRÌNH DỰ THI**  
**GIẢI THƯỞNG SINH VIÊN NGHIÊN CỨU KHOA HỌC EURÉKA**  
**LẦN THỨ XIX NĂM 2017**

TÊN CÔNG TRÌNH:

**THIẾT BỊ** **PHÂN TÍCH, ĐÁNH GIÁ CHẤT LƯỢNG ANTENNA TRONG CÁC THIẾT BỊ IoT**

LĨNH VỰC NGHIÊN CỨU: **CÔNG NGHỆ THÔNG TIN**

CHUYÊN NGÀNH: **ĐIỆN TỬ ỨNG DỤNG**

Mã số công trình: …………………………….

Mã số công trình: …………………………….

(Phần này do BTC Giải thưởng ghi)

***Ghi chú:***(Không ghi tên trường, tên sinh viên thực hiện, giáo viên hướng dẫn ở trang bìa và trong toàn bộ các trang của đề tài nghiên cứu. Đề nghị sinh viên ghi rõ công trình nghiên cứu thuộc lĩnh vực, nhóm ngành nào theo thể lệ của Ban tổ chức.)

# **MỞ ĐẦU**

Hiện nay, lĩnh vực IoT ở Việt Nam phát triển nhanh chóng, do đó yêu cầu về chất lượng của các sản phẩm IoT không chỉ ở phần mềm ổn định mà còn về khả năng giao tiếp với nhau, đặc biệt là giao tiếp không dây. Từ đó, nhu cầu về việc phân tích và đánh giá chất lượng của antenna trong sản phẩm IoT càng cần thiết. Tuy nhiên, các giải pháp cũng như thiết bị về việc phân tích, đánh giá chất lượng antenna còn ít và có nhiều hạn chế đặc biệt là ở thị trường Việt Nam.

Ở nước ta hiện nay, các thiết bị dùng để đo đạt, phân tích, đánh giá chất lượng antenna trong các thiết bị IoT đối với nhiều người vẫn còn khá xa lạ và mới mẻ, vì trong nước chưa có một cơ quan tổ chức hay cá nhân nào thực hiện đề tài này.

Nội dung của đề tài nghiên cứu này trình bày cách thức tiếp cận và xây dựng “Thiết bị dùng để đo đạt, phân tích, đánh giá chất lượng antenna” sử dụng Kit phát triển Renesas GR-Peach.

MỤC LỤC

[MỞ ĐẦU 2](#_Toc468533813)

[MỤC LỤC 3](#_Toc468533814)

[CHƯƠNG 1: GIỚI THIỆU 5](#_Toc468533815)

[1.1. Giới thiệu về Bubble Graphic Display 5](#_Toc468533816)

[1.2. Nội dung, mục tiêu và giới hạn của đề tài 6](#_Toc468533817)

[*1.2.1. Nội dung, mục tiêu đề tài* 6](#_Toc468533818)

[*1.2.2. Giới hạn đề tài* 6](#_Toc468533819)

[CHƯƠNG 2: KIẾN THỨC NỀN TẢNG 7](#_Toc468533820)

[2.1. Đặt vấn đề 7](#_Toc468533821)

[2.2. Vi điều khiển STM32F4 và một số linh kiện điện tử 7](#_Toc468533822)

[*2.2.1. Vi điều khiển STM32F4* 7](#_Toc468533823)

[2.2.1.1. Giới thiệu về Vi điều khiển STM32F4 7](#_Toc468533824)

[2.2.1.2. Các tính năng của STM32F4 9](#_Toc468533825)

[2.2.1.3. Giới thiệu KIT STM32F4 DISCOVERY 10](#_Toc468533826)

[*2.2.2. IC dịch 595* 11](#_Toc468533827)

[2.2.2.1. Giới thiệu về IC dịch 595 11](#_Toc468533828)

[2.2.2.2. Cấu tạo và hoạt động của IC dịch 595 12](#_Toc468533829)

[2.2.2.3. Sơ đồ chân và chức năng 12](#_Toc468533830)

[*2.2.3. MOSFET* 13](#_Toc468533831)

[2.2.3.1. Giới thiệu về MOSFET 13](#_Toc468533832)

[2.2.3.2. Cấu tạo và kí hiệu MOSFET 14](#_Toc468533833)

[2.2.3.3. Nguyên tắc hoạt động 14](#_Toc468533834)

[*2.2.4. Giao tiếp Vi điều khiển - PC theo chuẩn nối tiếp RS232* 15](#_Toc468533835)

[2.3. Kiến thức cơ bản về ảnh số và xử lý ảnh 16](#_Toc468533836)

[*2.3.1. Giới thiệu* 16](#_Toc468533837)

[*2.3.2. Lý thuyết ảnh màu cơ bản* 17](#_Toc468533838)

[2.3.2.1. Biểu diễn màu (Color Representation) 17](#_Toc468533839)

[2.3.2.2. Không gian màu (Color Space) 17](#_Toc468533840)

[2.3.2.3. Biểu diễn ảnh (Image Representation) 19](#_Toc468533841)

[*2.3.3. Các cách thức chuyển đổi ảnh cơ bản* 20](#_Toc468533842)

[2.3.3.1. Chuyển đổi ảnh màu sang ảnh xám 20](#_Toc468533843)

[2.3.3.2. Chuyển đổi ảnh xám sang ảnh nhị phân 21](#_Toc468533844)

[2.4. Tổng kết chương 21](#_Toc468533845)

[CHƯƠNG 3: HỆ THỐNG BUBBLE GRAPHIC DISPLAY 22](#_Toc468533846)

[3.1. Phân tích yêu cầu hệ thống 22](#_Toc468533847)

[3.2. Thiết kế tổng quan hệ thống 24](#_Toc468533848)

[CHƯƠNG 4: THIẾT KẾ MẠCH VÀ XÂY DỰNG CHƯƠNG TRÌNH ĐIỀU KHIỂN HỆ THỐNG BUBBLE GRAPHIC DISPLAY 25](#_Toc468533849)

[4.1. Thiết kế mạch điều khiển 25](#_Toc468533850)

[*4.1.1. Sơ đồ khối hệ thống và sơ đồ khối Kit STM32F4 Discovery* 25](#_Toc468533851)

[*4.1.2. Khối mạch Công suất* 27](#_Toc468533852)

[*4.1.3. Khối mạch 595* 30](#_Toc468533853)

[4.2. Xây dựng chương trình xử lý ảnh và điều khiển 31](#_Toc468533854)

[*4.2.1. Sơ đồ khối* 31](#_Toc468533855)

[*4.2.2. Chương trình xử lý ảnh trên máy tính* 32](#_Toc468533856)

[4.2.2.1. Giới thiệu 32](#_Toc468533857)

[4.2.2.2. Phần xử lý ảnh 32](#_Toc468533858)

[*4.2.2.2.1 Sơ đồ giải thuật* 32](#_Toc468533859)

[*4.2.2.2.2. Chuyển đổi kích thước ảnh theo tỷ lệ* 33](#_Toc468533860)

[*4.2.2.2.3. Chuyển đổi ảnh màu sang ảnh nhị phân* 34](#_Toc468533861)

[*4.2.2.2.4. Chuyển đổi ảnh nhị phân sang mảng 2 chiều* 35](#_Toc468533862)

[4.2.2.3. Phần truyền dữ liệu qua cổng COM 36](#_Toc468533863)

[*4.2.3. Chương trình điều khiển trên STM32F4* 40](#_Toc468533864)

[4.2.3.1. Giới thiệu 40](#_Toc468533865)

[4.2.3.2. Xây dựng chương trình điều khiển cho STM32F4 40](#_Toc468533866)

[4.3. Thiết kế lắp đặt dàn Solenoid 42](#_Toc468533867)

[4.4. Kết quả thực nghiệm thu được 43](#_Toc468533868)

[*4.4.1. Hệ thống mạch điều khiển* 43](#_Toc468533869)

[*4.4.2. Chương trình xử lý ảnh* 44](#_Toc468533870)

[*4.4.3. Kết quả thực nghiệm* 46](#_Toc468533871)

[4.5. Đánh giá 47](#_Toc468533872)

[5.1. Kết luận 48](#_Toc468533873)

[5.2. Hướng phát triển 48](#_Toc468533874)

[TÀI LIỆU THAM KHẢO 49](#_Toc468533875)

# 

# **CHƯƠNG 1: GIỚI THIỆU**

## **1.1. Giới thiệu về VNA – Vector Network Analyzer**

Hiện nay trên thế giới có rất nhiều loại thiết bị cũng như công nghệ dùng để phân tích, đánh giá chất lượng antenna trong các thiết bị IoT như: Voltage Standing Wave Ratio Meter (SWR Meter or VSWR Meter), Scalar Network Analyzer (SNA), Vector Network Analyzer (VNA), System on Chip Vector Network Analyzer,…

## 1. Voltage Standing Wave Ratio Meter (SWR Meter or VSWR Meter)

## • SWR Meter là thiết bị đo tỉ số sóng đứng (Standing Wave Ratio) trong đường truyền. Nếu không có phản xạ từ thiết bị tải (antenna) về đường truyền thì tỉ số sóng đứng bằng 1, tỉ số này càng lớn khi độ phản xạ càng nhiều. SWR Meter chỉ ra độ chênh lệch của sóng trên đường truyền và trên thiết bị tải (antenna) từ đó đưa ra đánh giá về chất lượng antenna.

## • Ưu điểm của SWR Meter là áp dụng phần mềm nhúng để đơn giản hóa phần cứng giúp giảm giá thành của thiết bị.

## • Khuyết điểm của SWR Meter là không thể đo được tổng trở thực trên thiết bị tải (antenna) mà chỉ đo được tỉ số sai lệch (Mismatch Ratio). Ngoài ra, việc áp dụng phần mềm nhúng khiến SWR Meter chỉ hoạt động được ở tần số thấp (khoảng dưới 600MHz).

## 2. Scalar Network Analyzer (SNA)

## • SNA là một loại thiết bị được dùng để phân tích, đánh giá chất lượng antenna dựa trên sóng phản xạ. Nguyên lý hoạt động của SNA tương tự như một máy phân tích phổ sóng (Spectrum Analyzer) kết hợp với một máy phát sóng ở tần số chỉ định (Tracking Generator) nhưng chỉ phân tích được các thuộc tính biên độ của sóng.

## • Ưu điểm chính của SNA là phần cứng dùng để phân tích chất lượng antenna và bộ đo năng lượng tương đối đơn giản, giá thành thấp. Vì kiến trúc phần cứng đơn giản, SNA có khả năng quét tầng số nhanh hơn các loại RF Network Analyzer khác.

## • Khuyết điểm của SNA là do bộ thu băng thông rộng và đơn giản nên rất nhạy cảm với tạp âm và nhiễu. Ngoài ra, SNA chỉ đo được biên độ sóng.

## 3. Vector Network Analyzer (VNA)

## • VNA là một thiết bị phân tích, đánh giá chất lượng antenna. Nguyên lý hoạt động của VNA giống với SNA. Tuy nhiên ngoài biên độ (amplitude), VNA còn đo được pha (phase) của sóng.

## • Có 2 loại VNA chính:

##  VNA thông thường:

## - Ưu điểm: Băng thông rộng, Phổ biến

## - Nhược điểm: Giá thành cao, Kích thước lớn

##  VNA với SoC (System on Chip) :

## - Ưu điểm: Băng thông rộng, Kích thước nhỏ (tích hợp trên một IC)

## - Nhược điểm: Sử dụng công nghệ vi mạch phức tạp, Chi phí thiết kế sản xuất tốn kém, Vẫn còn đang trong quá trình nghiên cứu

## Giá thành Kích thước Băng thông hoạt động

## SWR Meter Thấp Nhỏ Thấp (Dưới 600MHz)

## SNA Thấp Lớn Cao

## VNA Thông thường Cao Lớn Cao

## VNA với SoC Chưa có trên thị trường Nhỏ Cao (4GHz – 32GHz)

## Bảng 1: So sánh các thiết bị đánh giá chất lượng antenna.

## Sau khi tiến hành tìm hiểu ưu điểm và nhược điểm của các giải pháp cũng như thiết bị về phân tích, đánh giá chất lượng antenna, nhóm nghiên cứu quyết định nghiên cứu, chế tạo một thiết bị có những ưu điểm của các thiết bị đã đề cập và khác phục được những nhược điểm của các thiết bị này. Cụ thể thiết bị có những đặc điểm sau:

## • Giá thành: Thấp (dưới 100$)

## • Kích thước: Nhỏ

## • Băng thông hoạt động: Đáp ứng được các chuẩn kết nối không dây phổ biến trong IoT ở Việt Nam (ISM band 433MHz, 900MHz, 2.4GHz)

## • Thao tác: Dễ sử dụng.

## **1.2. Nội dung, mục tiêu và giới hạn của đề tài**

### *1.2.1. Nội dung, mục tiêu đề tài*

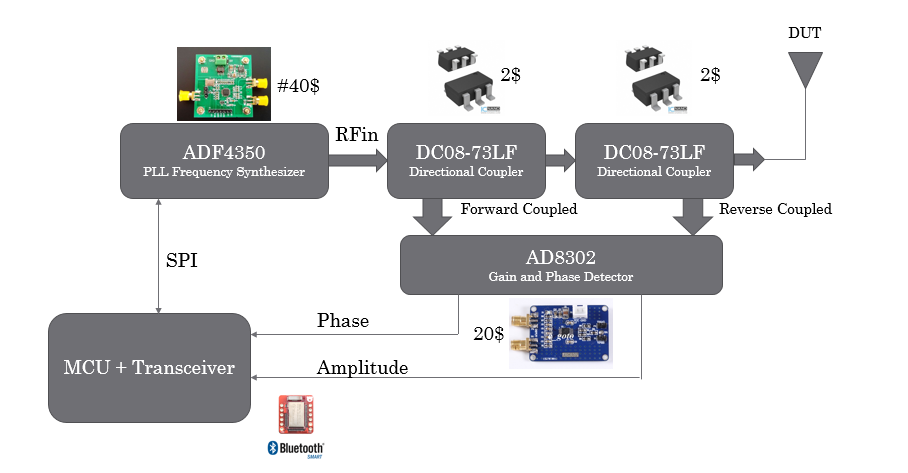
### Nghiên cứu, chế tạo thiết bị phân tích, đánh giá chính xác antenna với giá rẻ, gọn nhẹ và phù hợp cho thị trường IoT trong nước, từ đó giúp cho sản phẩm IoT Việt Nam có thể cải thiện được chất lượng và hiệu suất hơn. Thiết bị bao gồm các module sau:

### PLL Frequency Synthesizer: gồm có IC ADF4350 tạo ra sóng RF để thực hiện việc đo đạc.

### Directional Coupler: gồm 2 con IC DC0873 là nơi tách ra 2 sóng tới và sóng phản xạ, từ đó ta có thể so sánh chúng qua đó đánh giá được chất lượng của antenna tại DUT (Device under Test).

### Gain and Phase Detector: gồm IC AD8302 nơi tiếp nhận sóng tới và sóng phản xạ để so sánh rồi từ đó xuất ra được biên độ và biểu đồ trạng thái của sóng RF.

### MCU + Transceiver: Thông qua giao thức ADC (Analog to Digital Converter), MCU lấy dữ liệu về Phase và Amplitude từ module Gain and Phase Detector. Từ 2 thông số này và tầng số phát ra ở module PLL Frequency Synthesizer có thể phân tích được mức độ phản xạ sóng trên Antenna (DUT). Từ đó có thể đánh giá chất lượng của thiết kế antenna. Hiển thị lên LCD hoặc đưa ra thiết bị máy tính thông qua bluetooth để vẽ giản đồ chi tiết.



**Hình 1**: sơ đồ khối của hệ thống

### *1.2.2. Giới hạn đề tài*

Để xây dựng một THIẾT BỊ PHÂN TÍCH, ĐÁNH GIÁ CHẤT LƯỢNG ANTENNA TRONG CÁC THIẾT BỊ IoT hoàn chỉnh là một vấn đề lớn và tốn khá nhiều chi phí và thời gian bên cạnh đó do một số thiết bị (chẳng hạn như bộ phát tần số, bộ thu và so sánh tần số ) ở Việt Nam, cũng như các thị trường lần cận chất lượng chưa cao cho nên độ chính xác chưa cao như các thiết bị chuyên dụng. Do đó đề tài chỉ dừng lại ở việc xây dựng một thiết bị “*Vector Network Analyzer*” căn bản điều khiển trực tiếp bởi thiết bị di động ( cụ thể là điện thoại di động sử dụng hệ điều hành Android), thiết bị di động sẽ điều khiển và gửi lệnh xuống Renasas GR-Peach để điều khiển các mô-đun hoạt động, phân tích và hiển thị các giá trị lên màn hình LCD. Còn các khía cạnh nâng cao như điều khiển thiết bị “*Vector Network Analyzer*” qua Internet hoặc hiển thị trực tiếp trên Web … sẽ được nghiên cứu về sau.

# **CHƯƠNG 2: KIẾN THỨC NỀN TẢNG**

## **2.1. Đặt vấn đề**

Hệ thống “Mini *Vector Network Analyzer*” là sự kết hợp chặt chẽ giữa lập trình, mạch điện tử và van khí điện từ. Đây là ba yếu tố quan trọng chính để xây dựng một hệ thống “*Vector Network Analyzer*” căn bản.

Ý tưởng thiết kế mạch cho hệ thống “Mini *Vector Network Analyzer*” dựa trên hoạt động của thiết bị VNA chuyên dụng. **Mạch quang báo sử dụng IC dịch để điều khiển một số lượng lớn LED. Màn hình nước cũng vậy, hệ thống cũng phải điều khiển đóng mở một số lượng lớn van điện từ. Chỉ có sự khác biệt là van Solenoid có công suất lớn hơn rất nhiều so với LED. Do đó ta cần tìm hiểu các linh kiện để thiết kế mạch công suất để điều khiển trực tiếp các van này. Để giao tiếp giữa máy tính và vi điều khiển ta vẫn sử dụng chuẩn RS232, một chuẩn rất thông dụng trong điều khiển.**

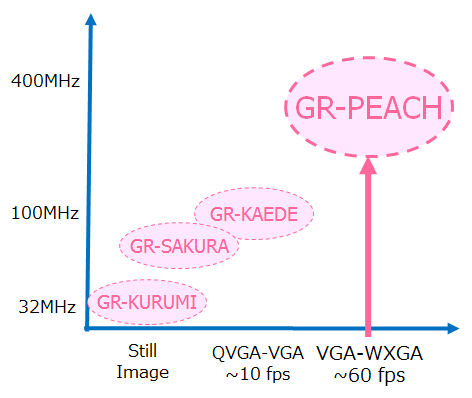
Về mảng lập trình, ta tìm hiểu một số kiến thức về hiển thị LCD 5110, điều khiển mô-đun Bluetooth HC-05 truyền dữ liệu và xử lý dữ liệu nhận được, hiển thị dưới dạng đồ thị trên thiết bị di động chạy hệ điều hành Android. Do môi trường Java hỗ trợ khá đầy đủ các thư viện và cũng là ngôn ngữ chính thống được Google hỗ trợ để phát triển ứng dụng trên thiết bị Android nên Java là ngôn ngữ được chọn để phát triển ứng dụng điều khiển và hiển thị kết quả đo đạt.

## **2.2. Kit phát triển Renesas GR-Peach và một số linh kiện điện tử**

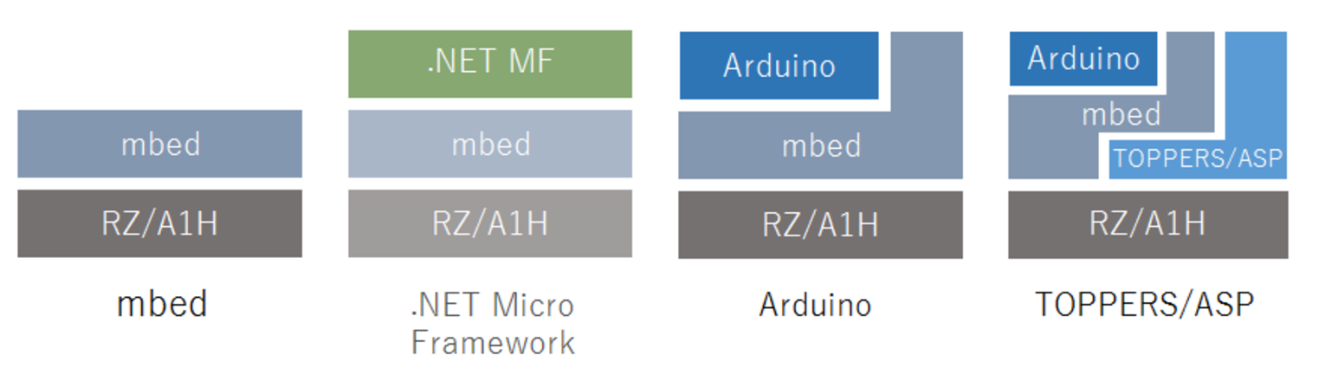
### *2.2.1* *Kit phát triển Renesas GR-Peach*

#### 2.2.1.1. Giới thiệu về Kit phát triển Renesas GR-Peach

Cấu trúc ARM (viết tắt từ tên gốc là Acorn RISC Machine) là một loại cấu trúc vi xử lý 32-bit kiểu RISC được sử dụng rộng rãi trong các thiết kế di động. Do có đặc điểm tiết kiệm năng lượng, các CPU ARM chiếm ưu thế trong các sản phẩm điện tử di động, mà với các sản phẩm này việc tiêu tán công suất thấp là một mục tiêu thiết kế quan trọng hàng đầu. Không giống như các chip ARM khác, dòng Cortex là một lõi xử lí hoàn thiện, đưa ra một chuẩn CPU và kiến trúc hệ thống chung. Dòng Cortex gồm có 3 phân nhánh chính: dòng A dành cho các ứng dụng cao cấp, dòng R dành cho các ứng dụng thời gian thực như các đầu đọc và dòng M dành cho các ứng dụng vi điều khiển và chi phí thấp. Kit phát triển Renesas GR-Peach được thiết kế dựa trên dòng Cortex-A9, dòng Cortex-A được thiết kế đặc biệt để nâng cao hiệu suất hệ thống, kết hợp với tốc độ xử lý cao. Cortex A9 được thiết kế trên nền công nghệ kiến trúc mới, với số lượng lỗi xử lý lên đến 4 lỗi cho tốc xử lý vượt trội so với các thế trước cũng như các kiến trúc xử lý khác trên thị trường.

****

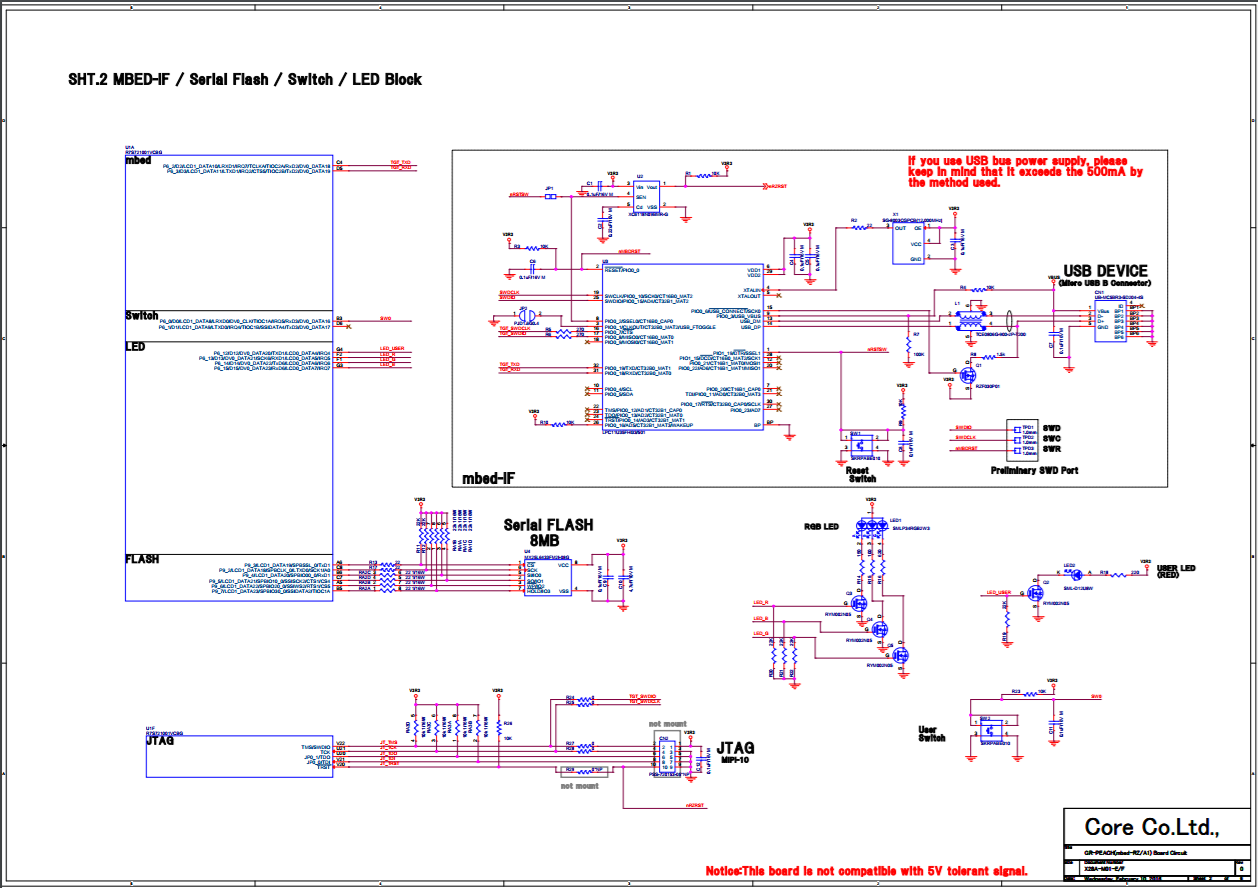
Hình 2.1: So sánh tốc độ giữa các Kit phát triển của Renesas

****

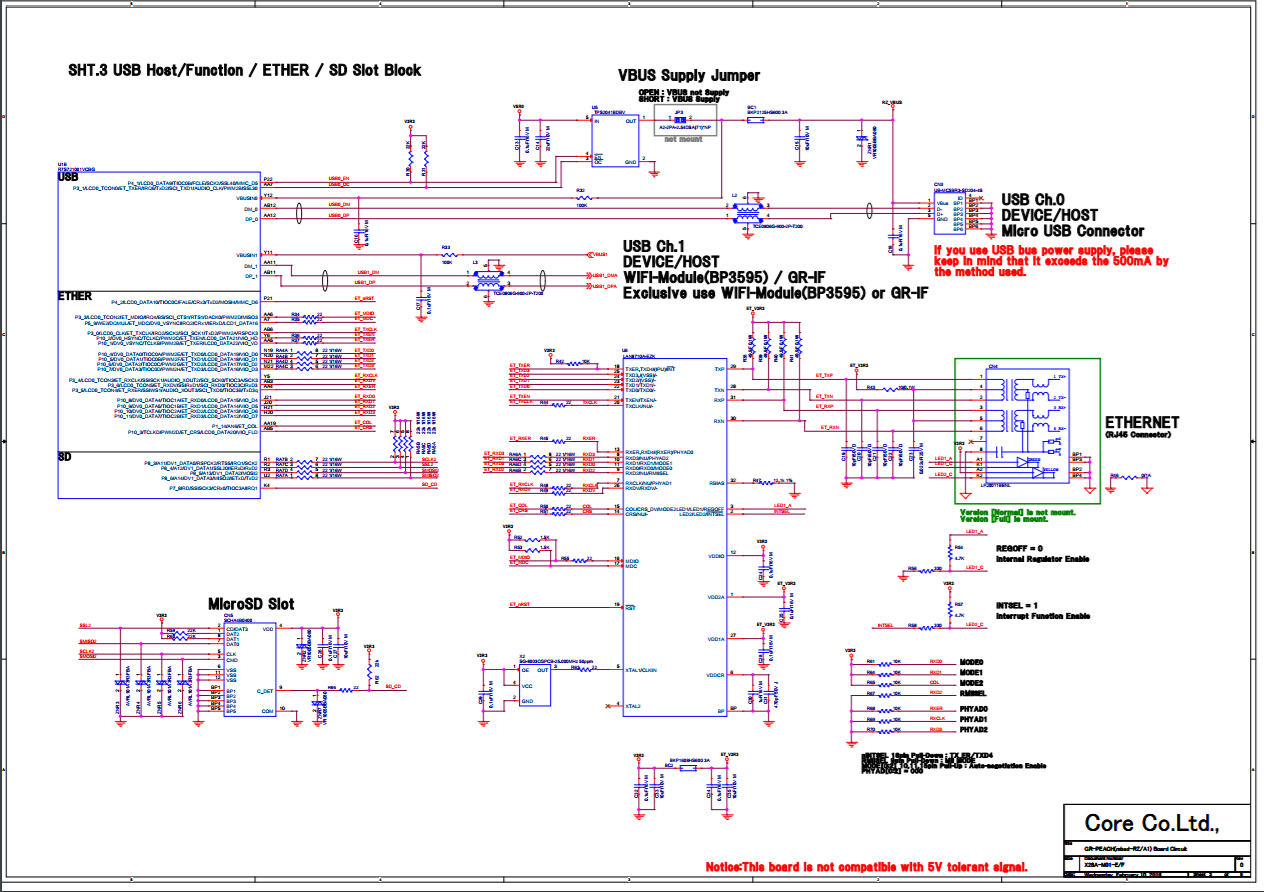
Hình 2.2: Các công nghệ có thể dùng để lập trình cho Kit phát triển

Renesas GR-Peach

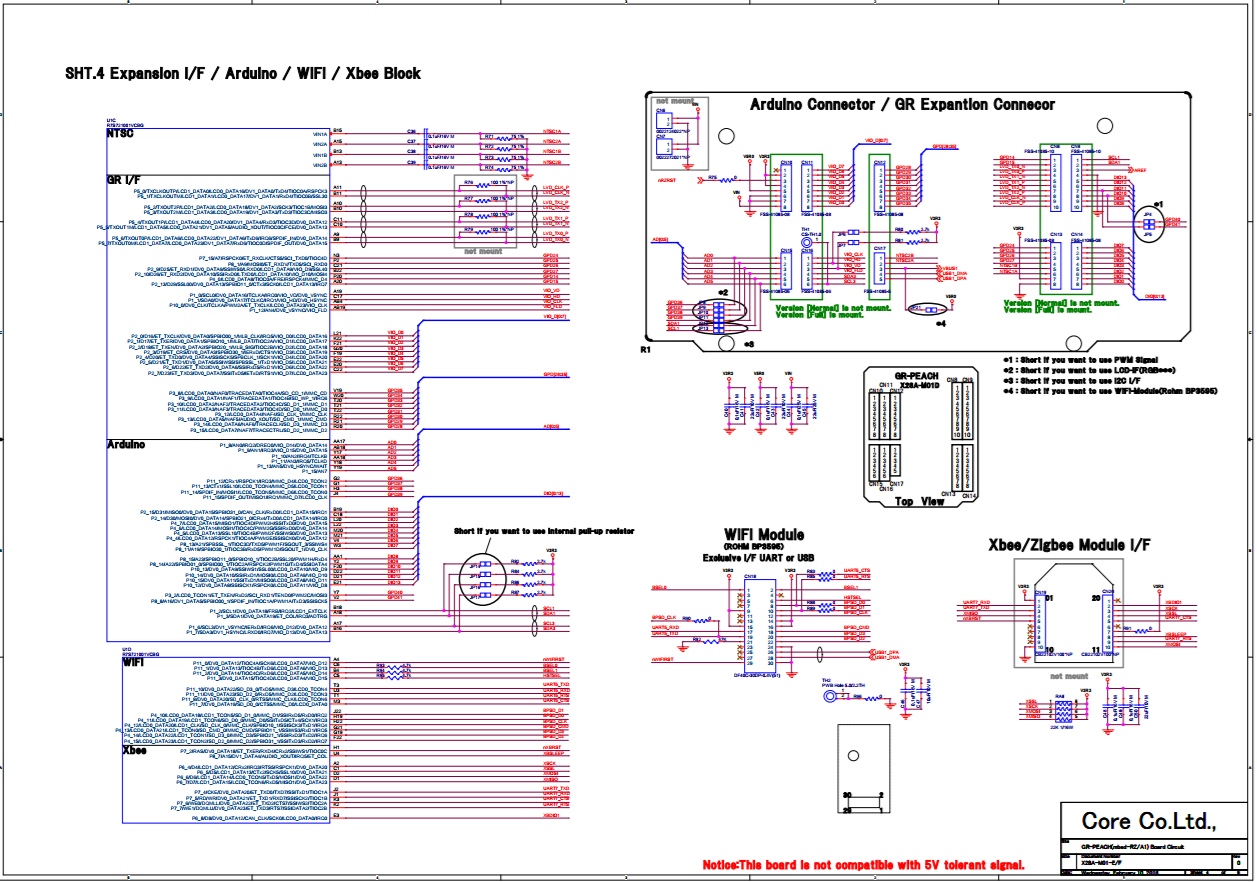
Renasas đã đưa ra thị trường rất nhiều kit phát triển dựa trên ARM7 và ARM9, nhưng Renasas có một bước tiến quan trọng trên đường cong chi phí và hiệu suất (price/performance), giá chỉ gần 1 Euro với số lượng lớn, Renasas là sự thách thức thật sự với các vi điều khiển 32-bit truyền thống.



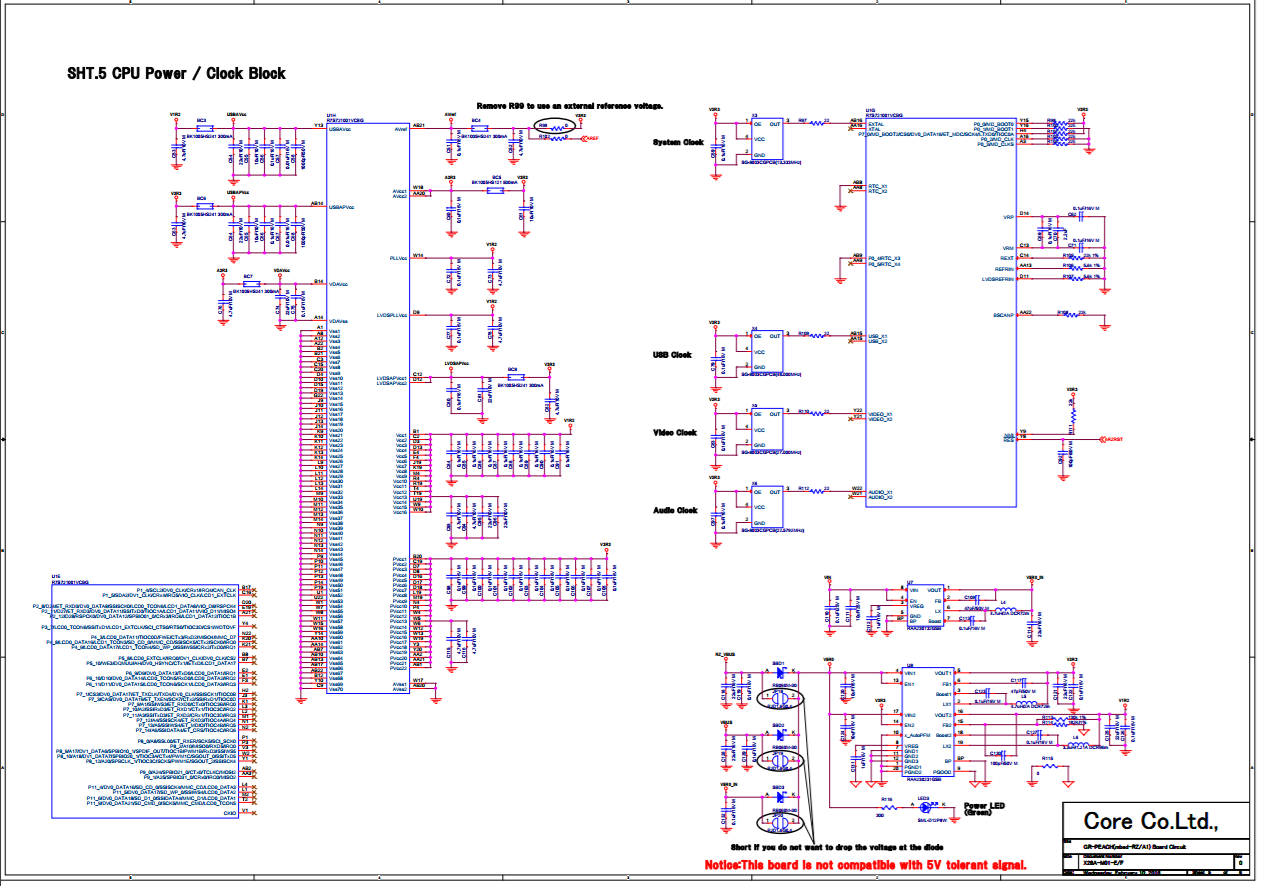
Hình 4.2: Sơ đồ Kit phát triển Renesas GR-Peach - 1



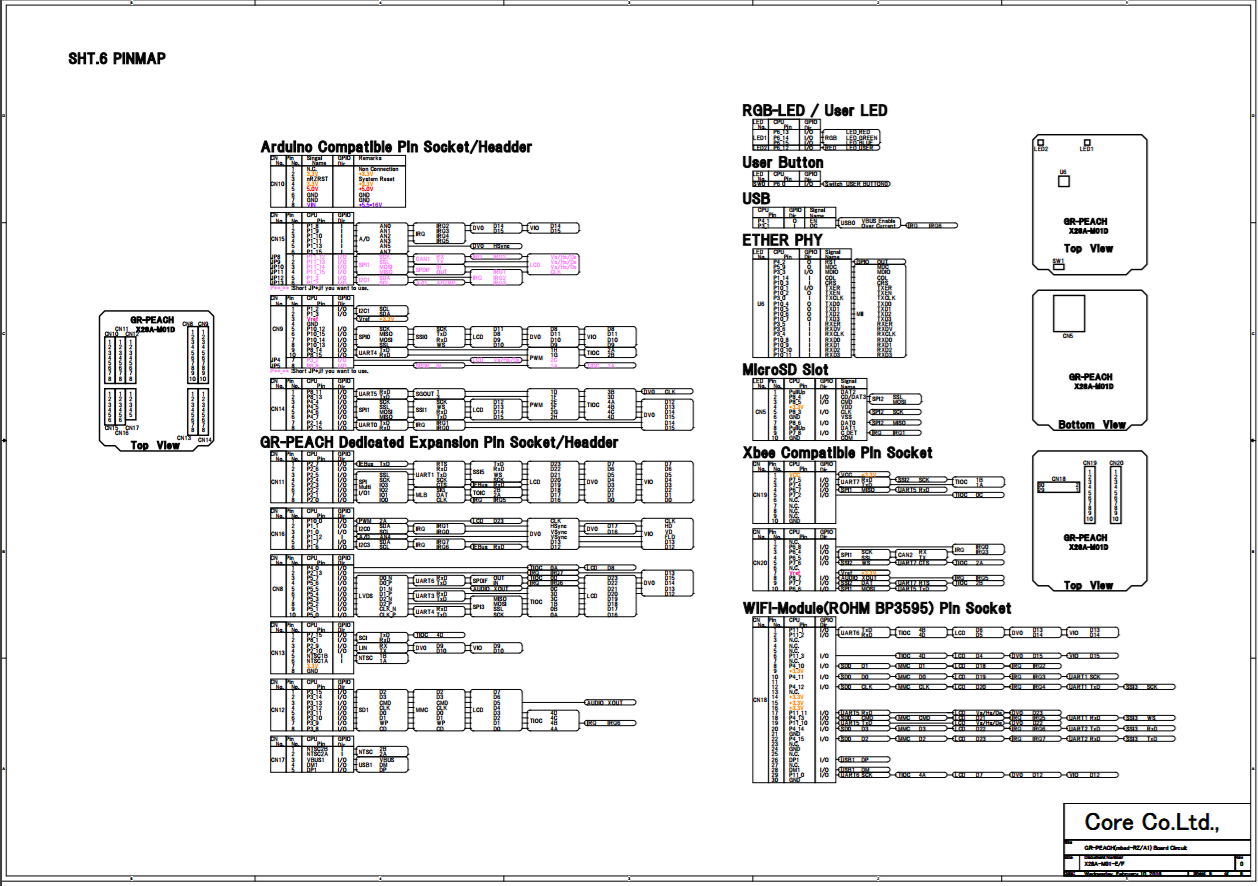
Hình 4.3: Sơ đồ Kit phát triển Renesas GR-Peach - 2



Hình 4.4: Sơ đồ Kit phát triển Renesas GR-Peach - 3



Hình 4.5: Sơ đồ Kit phát triển Renesas GR-Peach - 4



Hình 4.6: Sơ đồ Kit phát triển Renesas GR-Peach – 5

#### 2.2.1.2. Các tính năng của STM32F4

1. **Sự tinh vi**

Thoạt nhìn thì các ngoại vi của STM32 cũng giống như những vi điều khiển khác, như hai bộ chuyển đổi ADC, timer, I2C, SPI, CAN, USB và RTC. Tuy nhiên mỗi ngoại vi trên đều có rất nhiều đặc điểm thú vị. Ví dụ như bộ ADC 12-bit có tích hợp một cảm biến nhiệt độ để tự động hiệu chỉnh khi nhiệt độ thay đổi và hỗ trợ nhiều mode chuyển đổi. Mỗi bộ timer có 4 khối capture compare, mỗi khối timer có thể liên kết với các khối timer khác để tạo ra một mảng các timer tinh vi.

1. **Sự an toàn**

Ngày nay các ứng dụng hiện đại thường phải hoạt động trong môi trường khắc khe, đòi hỏi tính an toàn cao, cũng như đòi hỏi sức mạnh xử lý và càng nhiều thiết bị ngoại vi tinh vi. Để đáp ứng các yêu cầu khắc khe đó, STM32 cung cấp một số tính năng phần cứng hỗ trợ các ứng dụng một cách tốt nhất. Chúng bao gồm một bộ phát hiện điện áp thấp, một hệ thống bảo vệ xung clock và hai bộ watchdogs.

1. **Tính bảo mật**

Một trong những yêu cầu khắc khe khác của thiết kế hiện đại là nhu cầu bảo mật mã chương trình để ngăn chặn sao chép trái phép phần mềm. Bộ nhớ Flash của STM32 có thể được khóa để chống truy cập đọc Flash thông qua cổng debug. Khi tính năng bảo vệ đọc được kích hoạt, bộ nhớ Flash cũng được bảo vệ chống ghi để ngăn chặn mã không tin cậy được chèn vào bảng vector ngắt.

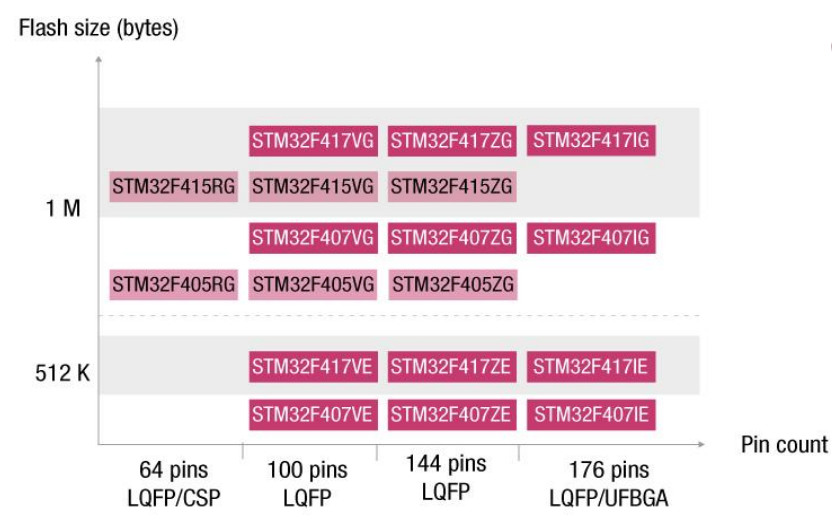
1. **Phát triển phần mềm**

Nếu như đã sử dụng một vi điều khiển dựa trên lõi ARM, các công cụ phát triển đã được hỗ trợ tập lệnh Thumb-2 và dòng Cortex. Ngoài ra ST cũng cung cấp một thư viện điều khiển thiết bị ngoại vi, một bộ thư viện phát triển USB như là một thư viện ANSI C và mã nguồn đó là tương thích với các thư viện trước đó được công bố cho vi điều khiển STR7 và STR9. Có rất nhiều RTOS mã nguồn mở và thương mại và middleware (TCP/IP, hệ thống tập tin, v.v.) hỗ trợ cho họ Cortex. Dòng Cortex-M3 cũng đi kèm với một hệ thống gỡ lỗi hoàn toàn mới gọi là CoreSight. Truy cập vào hệ thống CoreSight thông qua cổng truy cập Debug (Debug Access Port), cổng này hỗ trợ kết nối chuẩn JTAG hoặc giao diện 2 dây (serial wire-2 Pin), cũng như cung cấp trình điều khiển chạy gỡ lỗi.

1. **Dòng Performance và Access của STM32**

Họ STM32 có hai nhánh đầu tiên riêng biệt: dòng Performance và dòng Access. Dòng Performance tập hợp đầy đủ các thiết bị ngoại vi và chạy với xung nhịp tối đa 72MHz. Dòng Access có các thiết bị ngoại vi ít hơn và chạy tối đa 32MHz. Quan trọng hơn là cách bố trí chân (pins layout) và các kiểu đóng gói chip (package type) là như nhau giữa dòng Access và dòng Performance. Điều này cho phép các phiên bản khác nhau của STM32 được hoán vị mà không cần phải sửa đổi sắp sếp lại footprint (mô hình chân của chip trong công cụ layout bo mạch) trên PCB (Printed Circuit Board).

Ngoài hai dòng Performance và Access đầu tiên, hiện nay ST đã đưa ra thị trường thêm hai dòng USB Access và Connectivity như hình bên dưới.



Hình 2.3: Bộ nhớ và quy cách đóng gói dòng STM32F4

#### 2.2.1.3. Giới thiệu KIT STM32F4 DISCOVERY

Để thuận tiện cho việc học và nghiên cứu Vi xử lý ARM STM32, lựa chọn Kit Vi xử lý STM32F4 Discovery sẽ là sự thuận tiện cho việc học và nghiên cứu các chức năng cũng như việc lập trình các dòng ARM.

Ở đây hệ thống sẽ sử dụng vi điều khiển STM32F407VGT6, 100LQFP package

***Thông tin chung về sản phẩm:***

* Lõi: ARM Cortex M4F 32 bit.
* Bộ nhớ: 1MB Flash, 192 KB RAM.
* Tích hợp sẵn mạch nạp và debug ST-Link V2.
* Điện áp: sử dụng trực tiếp nguồn từ cổng USB, qua IC ổn áp 3.3V để nuôi mạch.
* Có sẵn cảm biến gia tốc: LIS302DL, ST MEMS motion sensor, 3-axis.
* Có sẵn bộ xử lý âm thanh: MP45DT02, ST MEMS audio sensor, omni-irectional digital microphone.
* Có sẵn bộ: CS43L22, audio DAC with integrated class D speaker driver.
* Có cổng Micro USB OTG.
* Tổng số I/O:
* ADC: 3x12 bit, 24 kênh 2.44MSPS
* DAC: 2x12 bit, 2 kênh
* DMA: Điều khiển 16 kênh DMA
* Timer: 10x16 bit, 2x32 bit
* Giao diện kết nối: 3xI2C, 6xUSART, 3xSPI, 2xCAN, 1xUSB 2.0 OTG FS/HS, 1x USB 2.0 OTG FS, Ethernet MAC 10/100 with IEEE 1588, …
* Kiểu chân: LQFP100

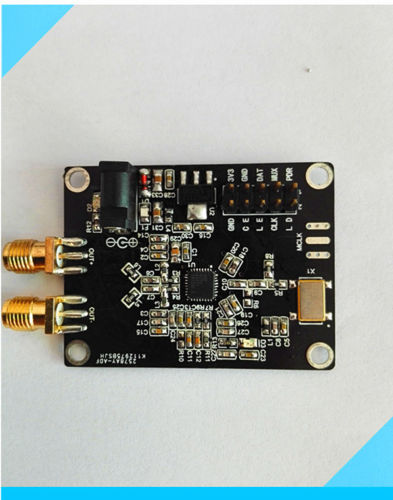


Hình 2.4: Kit STM32F4 Discovery

### *2.2.2. Mô-đun ADF4351*

#### 2.2.2.1. Giới thiệu về Mô-đun ADF4351

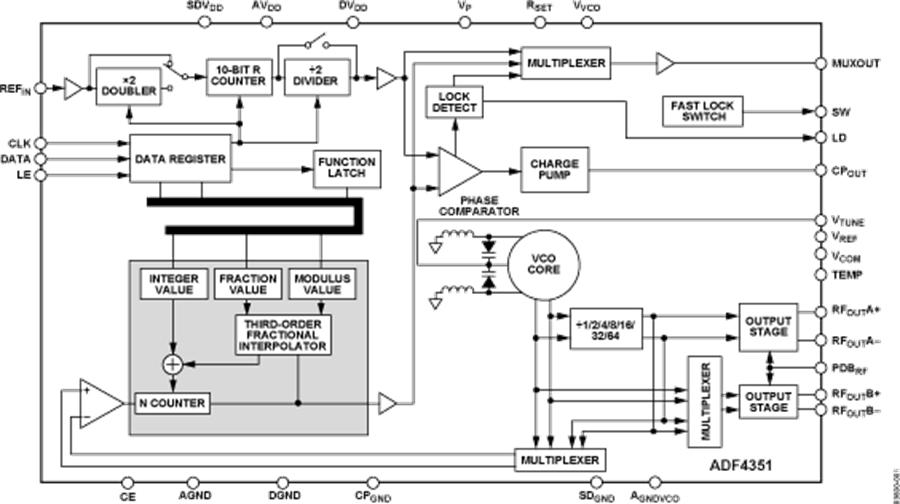
Khối mô-đun ADF4351 Evaluation Board là khối trực tiếp thực hiện việc tạo ra sóng ở tần số nhất định phụ thuộc yêu cầu phân tích, đánh giá antenna của người sử dụng thiết bị. IC chính trên mô-đun ADF4351 Evaluation Board là ADF4351 do nhà sản xuất Analog Devices sản xuất.



Hình 2.5: Mô-đun ADF4351

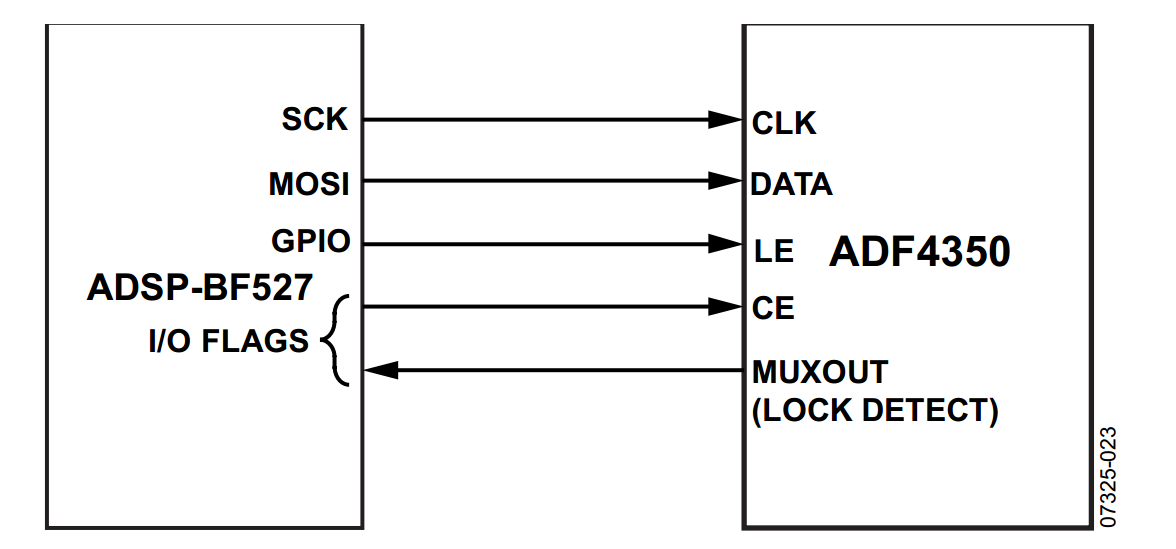
#### 2.2.2.2. Cấu tạo và hoạt động của Mô-đun ADF4351

Sơ đồ khối của Mô-đun ADF4351 được mô tả như trong Hình 2.6:



Hình 2.6: Sơ đồ khối IC ADF4351

Cách hoạt động của Mô-đun ADF4351được mô tả trực quan như trong Hình 2.7:

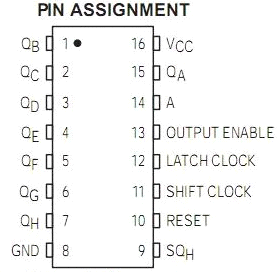


Hình 2.7:Nguyên lý giao tiếp IC dịch ADF4351

Bit được đưa vào chân DS (A), chân SH (SHIFT CLOCK) có nhiệm vụ cho phép bit đó có được phép ghi vào thanh ghi dịch hay không, chân ST (LATCH CLOCK) có nhiệm vụ cho phép bit xuất ra ngoài hay không. Một IC 595 lưu trữ được 8 bit, nếu bit thứ 9 được đưa vào thì nó sẽ đẩy bit ở chân SQH (bit vào đầu tiên) ra chân SQH, bit này sẽ làm bit đầu vào cho thanh ghi dịch tiếp theo. Thông thường khi sử dụng nhiều IC 595 điều khiển, ta thường nối chân SQH của IC 595 này với chân DS của IC 595 tiếp theo.

#### 2.2.2.3. Sơ đồ chân và chức năng

Sơ đồ các chân IC dịch 595 được thể hiện trong hình 2.8.



Hình 2.8: Sơ đồ chân IC 595

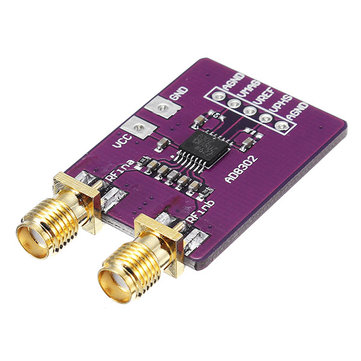
Giải thích ý nghĩa hoạt động của một số chân quan trọng:

* Chân 14 (Input): Đầu vào dữ liệu nối tiếp. Tại một thời điểm xung clock chỉ đưa vào được 1 bit.
* Các chân QA=>QH trên các chân (15, 1, 2, 3, 4, 5, 6, 7) (Output): Xuất dữ liệu khi chân 13 tích cực ở mức thấp và có một xung tích cực ở sườn âm tại chân chốt 12.
* Chân 13 (Output Enable): Chân cho phép tích cực ở mức thấp (0). Khi ở mức cao, tất cả các đầu ra của 74HC595 trở về trạng thái cao trở, không có đầu ra nào được cho phép.
* Chân 9 (SQH): Chân dữ liệu nối tiếp. Nếu dùng nhiều 74HC595 mắc nối tiếp nhau thì chân này đưa vào đầu của con tiếp theo khi đã dịch đủ 8 bit.
* Chân 11 (Shift Clock): Chân vào xung clock. Khi có 1 xung clock tích cực ở sườn dương (từ 0 lên 1) thì 1 bit được dịch vào IC.
* Chân 12 (Latch Clock): Chân vào xung clock chốt dữ liệu. Khi có 1 xung clock tích cực ở sườn dương thì cho phép xuất dữ liệu trên các chân output. Lưu ý có thể xuất dữ liệu bất kỳ lúc nào ta muốn, ví dụ đầu vào chân 14 được 2 bit khi có xung clock ở chân 12 thì dữ liệu sẽ ra ở chân QA và QB (chiều dịch dữ liệu từ QA=>QH).
* Chân 10 (Reset): Khi chân này ở mức thấp (mức 0) thì dữ liệu sẽ bị xóa trên chip.

### *2.2.3. Mô-đun AD8302*

#### 2.2.3.1. Giới thiệu về Mô-đun AD8302

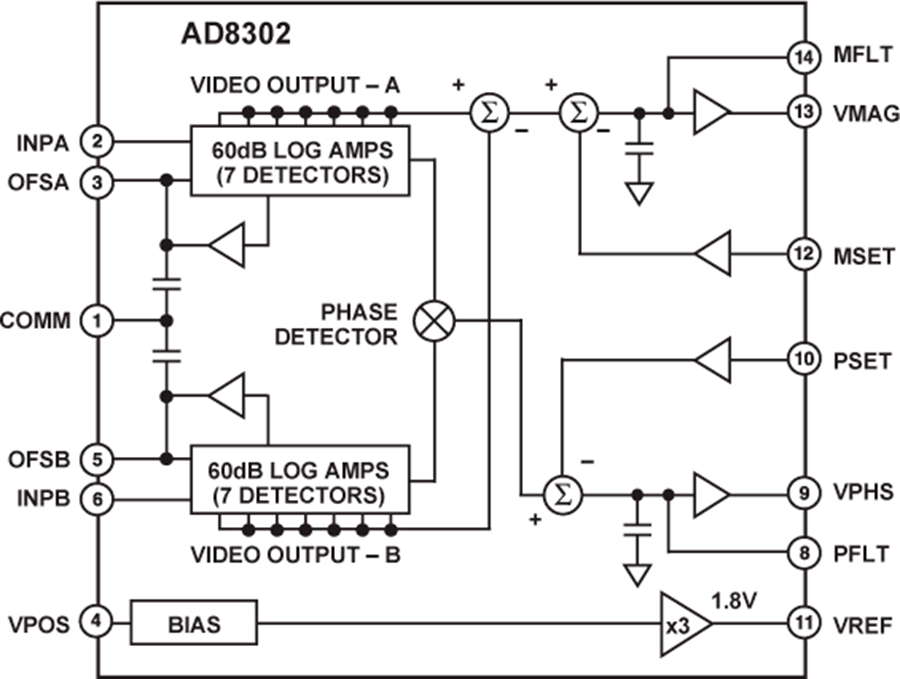
Metal Oxide Semiconductor Field Effect Transistor (MOSFET) là transistor hiệu ứng trường, một Transistor đặc biệt có cấu tạo và hoạt động khác với Transistor thông thường mà ta đã biết. Hình 2.9 thể hiện hình dáng và cấu tạo của một MOSFET trên thực tế. MOSFET có nguyên tắc hoạt động dựa trên hiệu ứng từ trường để tạo ra dòng điện, là linh kiện có trở kháng đầu vào lớn thích hợp cho khuyếch đại các nguồn tín hiệu yếu, MOSFET được sử dụng nhiều trong các mạch nguồn monitor, nguồn máy tính.



Hình 2.9: Mô-đun AD8302

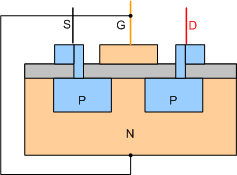
#### 2.2.3.2. Cấu tạo và kí hiệu Mô-đun AD8302

Ký hiệu Mosfet trong sơ đồ nguyên lý như sau:



Hình 2.10: Ký hiệu MOSFET

#### 2.2.3.3. Nguyên tắc hoạt động



Hình 2.11. Cấu tạo MOSFET

Cấu tạo của MOSFET ngược Kênh N gồm các thành phần sau:

* G: Gate gọi là cực cổng.
* S: Source gọi là cực nguồn.
* D: Drain gọi là cực máng.

Trong đó: G là cực điều khiển được cách ly hoàn toàn với cấu trúc bán dẫn còn lại bởi lớp điện môi cực mỏng nhưng có độ cách điện cực lớn dioxit-silic (SiO2). Hai cực còn lại là cực gốc (S) và cực máng (D). Cực máng là cực đón các hạt mang điện.

MOSFET có điện trở giữa cực G với cực S và giữa cực G với cực D là vô cùng lớn, còn điện trở giữa cực D và cực S phụ thuộc vào điện áp chênh lệch giữa cực G và cực S (UGS).

Khi điện áp UGS = 0 thì điện trở RDS rất lớn, khi điện áp UGS > 0 do hiệu ứng từ trường làm cho điện trở RDS giảm. Điện áp UGS càng lớn thì điện trở RDS càng nhỏ.

### *2.2.4. Giao tiếp Vi điều khiển - PC theo chuẩn nối tiếp RS232*

Vấn đề giao tiếp giữa vi điều khiển và máy tính rất quan trọng trong các ứng dụng điều khiển, đo lường … Ghép nối qua cổng nối tiếp RS232 là một trong những kỹ thuật được sử dụng rộng rãi để ghép nối các thiết bị ngoại vi với máy tính. Nó là chuẩn giao tiếp nối tiếp dùng đinh dạng không đồng bộ, kết nối nhiều nhất là 2 thiết bị, chiều dài kết nối lớn nhất cho phép đảm bảo dữ liệu là 12.5 đến 25.4 m, tốc độ 20 kbit/s. Ý nghĩa chuẩn truyền thông nối tiếp là trong một thời điểm chỉ có một bit được gửi đi dọc đường truyền.

Trên STM32F4 sử dụng truyền thông nối tiếp không đồng bộ (UART) thường để chỉ thiết bị phần cứng, không phải chỉ một chuẩn giao tiếp. Với chuẩn RS232 trên các máy tính cá nhân là sự kết hợp của chip UART (chẳng hạn như STM32F4) và chip chuyển đổi mức điện áp.

Tín hiệu từ chip UART thường theo mức TTL:

* Ngõ vào: - Mức thấp (0): 0V ─ +0.8V

- Mức cao (1): +2V ─ +5V

* Ngõ ra: - Mức thấp (0): 0 ─ +0.5V

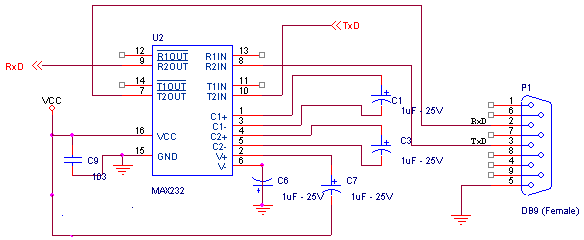
- Mức cao (1): +2V ─ +5V

Tín hiệu theo chuẩn RS232:

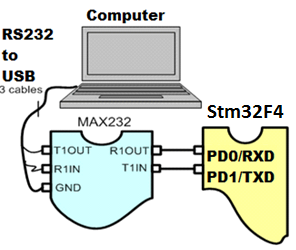
* Mức thấp (0): −12V ─ −3V
* Mức cao (1): +3V ─ +12V

Như thế, rõ ràng cần một “cầu chuyển” kết nối giữa 2 chuẩn này. Trên thị trường có loại IC chuyên dụng làm việc này. Đó là IC MAX232 chuyển UART-RS232 được sử dụng nhiều nhất.

Sơ đồ ghép nối với MAX232 được thể hiện trong hình sau:



Hình 2.12: Sơ đồ ghép nối MAX 232 với cổng COM



Hình 2.13: Kết nối RS232 với PC

## **2.3. Kiến thức cơ bản về ảnh số và xử lý ảnh**

### *2.3.1. Giới thiệu*

Để xây dựng 1 hệ thống Bubble Graphic Display thì vấn đề xử lý hình ảnh đóng vai trò quan trọng. Hệ thống yêu cầu phải xây dựng chương trình cho phép người dùng khi đưa một tấm ảnh màu với kích thước bất kì vào đều có thể hiển thị ra ngoài Màn hình bọt khí được.

Mục đích của việc xử lý ảnh trong chương trình điều khiển “bubble”:

* Chuyển ảnh màu sang ảnh nhị phân (ảnh trắng đen).
* Thay đổi kích thước ảnh theo tỉ lệ.
* Xử lý ảnh đen trắng trên pixels chuyển về mảng các giá trị 0, 1.

Để xây dựng 1 chương trình xử lý ảnh đơn giản, trước hết ta cần tìm hiểu thêm một số kiến thức cơ bản trong lập trình xử lý hình ảnh. Ở đây sử dụng ngôn ngữ C# và .NET framework.

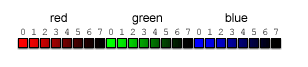
### *2.3.2. Lý thuyết ảnh màu cơ bản*

#### 2.3.2.1. Biểu diễn màu (Color Representation)

Màu sắc trên máy tính thưởng được biểu diễn bằng 2 cách thông dụng: 8-bit grayscale và 24-bit color.

Đối với thang màu xám 8-bit grayscale chứa 256 cấp độ của màu xám , trong đó 0 thể hiện cho màu đen và 255 thể hiện cho màu trắng ,các giá trị còn lại thể hiện các màu sắc khác nhau .

Còn đối với thang màu 24-bit là lưu 3 bytes thể hiện 3 thành phần màu bao gồm: đỏ (R), xanh (G) và xanh dương (B). Sự kết hợp của ba thành phần RGB có thể tạo nên một màu bất kỳ:



Hình 2.14: Biểu diễn màu [4]

Ngoài cách biểu diễn màu theo ba thành phần RGB còn có nhiều cách biểu diễn khác: HSI (Hue, Saturation, Intensity), CMY (Cyan, Magenta, Yellow), Normalized RG, CIE, YIQ …

#### 2.3.2.2. Không gian màu (Color Space)

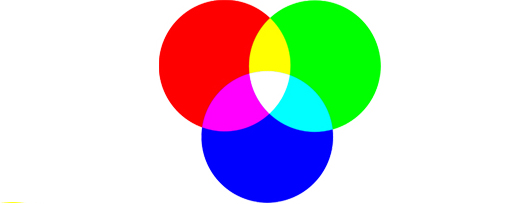
Không gian màu là một mô hình toán học dùng để mô tả các màu sắc trong thực tế được biểu diễn dưới dạng số học. Trên thực tế có rất nhiều không gian màu khác nhau được mô hình để sử dụng vào những mục đích khác nhau. Ba không gian màu cơ bản và được sử dụng phổ biến nhất sẽ được giới thiệu, bao gồm RGB, HSV, và CMYK.

**Không gian màu RGB:**

RGB là không gian màu rất phổ biến được dùng trong đồ họa máy tính và nhiều thiết bị kĩ thuật số khác. Ý tưởng chính của không gian màu này là sự kết hợp của 3 màu sắc cơ bản R, G, B để mô tả tất cả các màu sắc khác.

Nếu như một ảnh số được mã hóa bằng 24 bit, nghĩa là 8 bit cho kênh R, 8 bit cho kênh G, và 8 bit cho kênh B, thì mỗi kênh màu này sẽ nhận giá trị từ 0 đến 255. Với mỗi giá trị khác nhau của các kênh màu kết hợp với nhau ta sẽ được một màu khác nhau, như vậy ta sẽ có tổng cộng 2563 ≈ 1.66 triệu màu sắc.

Ví dụ: màu đen là sự kết hợp của các kênh màu (R, G, B) với giá trị tương ứng (0, 0, 0), màu trắng có giá trị (255, 255, 255), màu vàng có giá trị (255, 255, 0), màu tím đậm có giá trị (64, 0, 128).

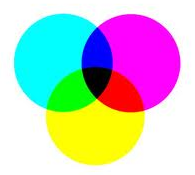


Hình 2.15: Không gian màu RBG [4]

**Không gian màu CMYK:**

CMYK là không gian màu được sử dụng phổ biến trong ngành công nghiệp in ấn. Ý tưởng cơ bản của hệ không gian này là dùng 4 màu sắc cơ bản để phục vụ cho việc pha trộn mực in. Trên thực tế, người ta dùng 3 màu là xanh lơ (Cyan – C), hồng xẫm (Magenta- M), và vàng (Yellow-Y) để biểu diễn các màu sắc khác nhau.

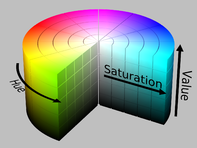
Ví dụ: Nếu lấy màu hồng xẫm cộng với vàng sẽ ra màu đỏ, màu xẫm kết hợp với xanh lơ sẽ cho xanh lam ... Sự kết hợp của 3 màu trên sẽ cho ra màu đen.



Hình 2.16: Không gian màu CYMK [4]

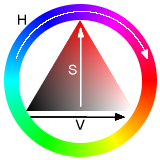
**Không gian màu HSV:**

HSV là không gian màu được dùng nhiều trong việc chỉnh sữa ảnh, phân tích ảnh và một phần của lĩnh vực thị giác máy tính. Hệ không gian này dựa vào 3 thông số sau để mô tả màu sắc: màu sắc (Hue - H), độ đậm đặc (Saturation - S), và giá trị cường độ sáng (Value - V). Không gian màu này thường được biểu diễn dưới dạng hình trụ hoặc hình nón như trong Hình 2.25:



Hình 2.17: Không gian màu HSV [4]

Không gian màu biểu diễn màu sắc (Hue) theo vòng tròn như hình 2.26 bên dưới:

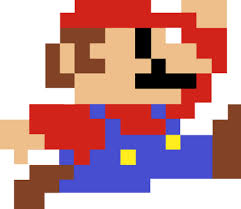


Hình 2.18: Hình tròn biểu diễn màu sắc (Hue) [4]

Như vậy với mỗi giá trị (H, S, V) sẽ cho ta một màu sắc mà ở đó mô tả đầy đủ thông tin về máu sắc, độ đậm đặc và độ sáng của màu đó.

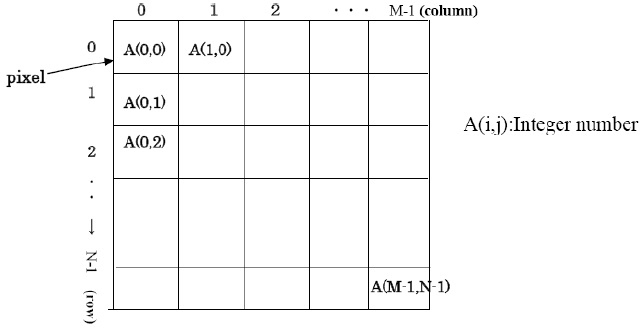
#### 2.3.2.3. Biểu diễn ảnh (Image Representation)

Ảnh số trên máy tính là một tập hợp các điểm ảnh nhỏ, mỗi điểm ảnh gọi là 1 pixe, mỗi điểm ảnh thể hiện một màu sắc nhất định và biểu diễn bằng một giá trị các bit.



Hình 2.19: Biểu diễn ảnh

Ví dụ trong hình 2.27, ảnh nhân vật Mario được tạo thành từ nhiều điểm ảnh khác nhau. Mỗi điểm ảnh được thể hiện bởi một màu sắc nhất định. Các điểm ảnh được sắp xếp như một mảng 2 chiều.

[](https://sites.google.com/site/dangtrieu/xulyanh/DigitalImage.jpg?attredirects=0)

Hình 2.20: Biểu diễn vị trí các pixel trên ảnh

**Phân loại: Có 3 loại ảnh:**

* Ảnh nhị phân (binary image): giá trị mỗi pixel là 0 hoặc 1, nghĩa là trắng hoặc đen. Trong thực tế khi xử lý trên máy tính thì người ta dùng ảnh xám (xem khái niệm bên dưới) để biểu diễn ảnh nhị phân và lúc này 2 giá trị là 0 hoặc 255.
* Ảnh xám (gray image): giá trị mỗi pixel nằm trong giải giá trị từ 0 đến 255, nghĩa là cần 8 bit hay 1 byte để biểu diễn mỗi pixel này.
* Ảnh màu (color image): mỗi pixel có giá trị gồm 3 RGB. Mỗi kênh màu có giá trị từ 0 đến 255, nghĩa là mỗi pixel cần 24 bit hay 3 byte để biểu diễn.



Ảnh màu  Ảnh xám Ảnh nhị phân

Hình 2.21: Các loại ảnh

### *2.3.3. Các cách thức chuyển đổi ảnh cơ bản*

#### 2.3.3.1. Chuyển đổi ảnh màu sang ảnh xám

Để chuyển từ ảnh màu sang ảnh xám, ta chỉ cần nhân giá trị các kênh màu của mỗi pixel với các hệ số cho trước:

|  |
| --- |
| *Y = 0.3\*Red + 0.59\*Green + 0.11\*Blue [1].* |

Xây dựng chương trình:

    + Kích thước của bức ảnh là X\*Y.

+ uchar c[3\*X][Y] là ma trận 2 chiều chứa dữ liệu của bức ảnh màu, 3 màu xếp theo thứ tự R, G, B.

    + uchar g[X][Y] là ma trận 2 chiều chứa dữ liệu của bức ảnh xám.

    + Duyệt từng pixels của cả bức ảnh (raster scan) bằng 2 vòng lặp và gán giá trị ảnh màu nhân với hệ số vào ảnh xám.

|  |
| --- |
| *for (i = 0; i <= X; i++) {*  *for (j = 0; j <= Y; j++) {*  *g[i][j] = 0.3\*c[3\*i][j] + 0.59\*c[3\*i+1][j] + 0.11\*c[3\*i+2][j];*  *} }* |

#### 2.3.3.2. Chuyển đổi ảnh xám sang ảnh nhị phân

Thao tác này được gọi là đặt ngưỡng, ta chỉ cần chọn một giá trị ngưỡng (threshold) thích hợp, pixel xám nào có giá trị bằng hoặc dưới ngưỡng đó ta cho bằng 0, trên ngưỡng đó ta cho bằng 255.

|  |
| --- |
| *if g[i][j] <= threshold         b[i][j] = 0; // b[i][j] là mảng 2 chiều chứa dữ liệu của bức ảnh nhị phân.     else         b[i][j] = 255;* |

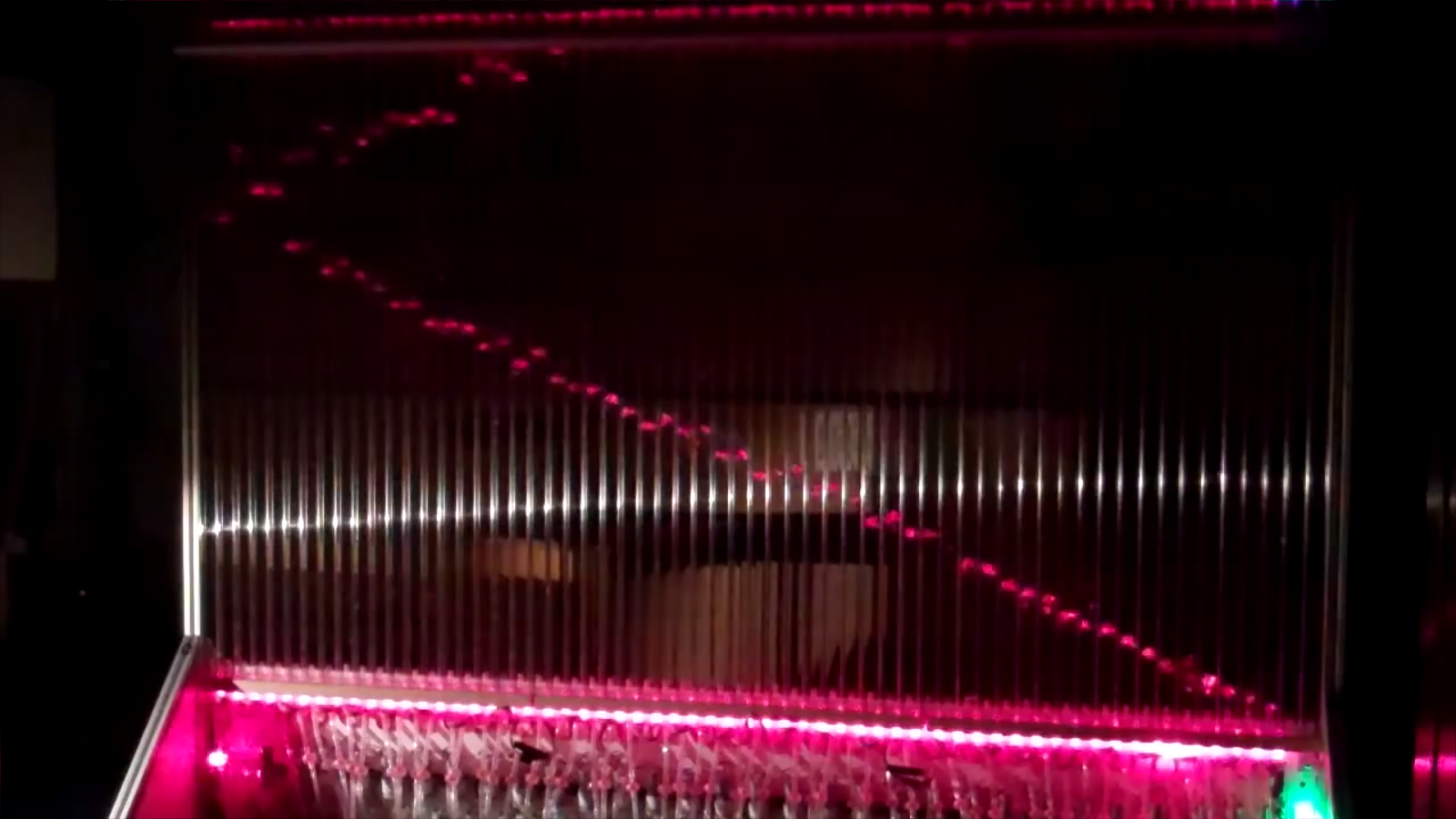
## **2.4. Tổng kết chương**

Như vậy, trong chương này đã đề cập đến một số linh kiện điện tử thông dụng sẽ được sử dụng để thiết kế mạch điều khiển và một số kiến thức căn bản hỗ trợ cho việc lập trình xử lý ảnh số trên máy tính. Đây là những kiến thức quan trọng làm nền tảng cho việc xây dựng một hệ thống Bubble Graphic Display.

# **CHƯƠNG 3: HỆ THỐNG BUBBLE GRAPHIC DISPLAY**

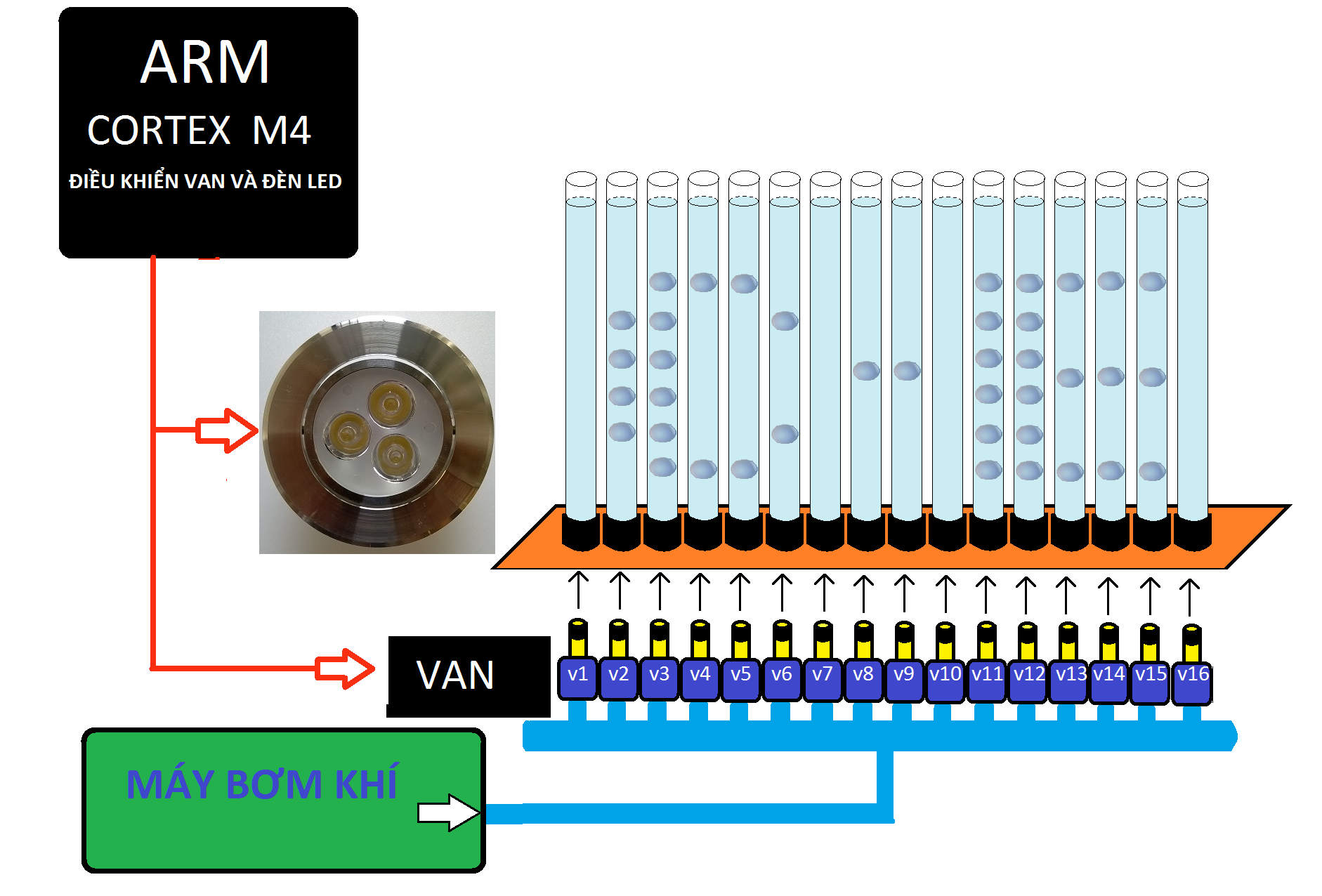
## **3.1. Phân tích yêu cầu hệ thống**

Bubble Graphic là hệ thống mà khi bọt khí nỗi tự do trong nước, các bọt khí kết hợp với nhau tạo thành hình ảnh, hiệu ứng …



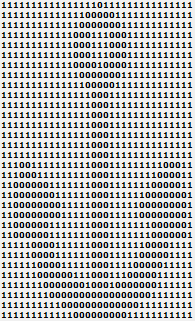
Hình 3.1: Bubble Graphic Display hoàn chỉnh

Các bọt khí được tạo bởi sự đóng mở liên tục của các van Solenoid (hay còn gọi là van điện tử). Các van Solenoid được đặt sát với nhau theo một hàng ngang, mỗi đầu van cách nhau khoảng 3cm. Các van này được điều khiển trực tiếp bởi một hệ thống mạch bên dưới. Khi van đóng mở ở tốc độ cao trong một khoảng thời gian ngắn (tính bằng ms) sẽ tạo ra bọt khí nỗi lên dưới áp lực đẩy khí từ máy bơm. Các bọt khí khi kết hợp lại sẽ tạo ra những hình ảnh, chữ số theo ý muốn.



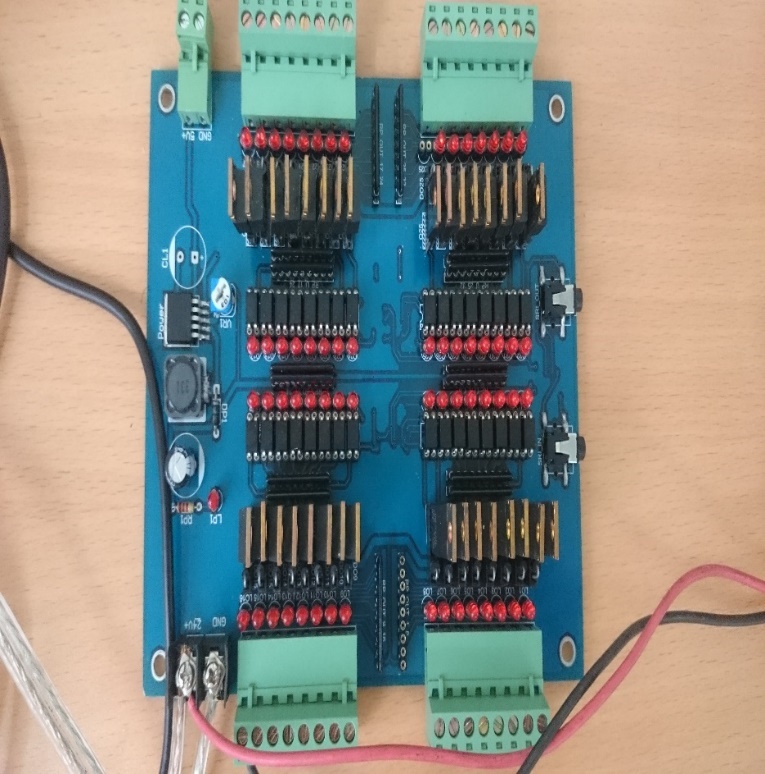
Hình 3.2: Hoạt động của van Solenoid

Ý tưởng xây dựng hệ thống dựa trên hoạt động của màn hình quang báo, hình ảnh được chuyển sang dạng nhị phân sau đó chuyển sang dạng mảng 2 chiều chỉ bao gồm các bit 0 và 1, những điểm ảnh có màu đen sẽ mang bit 0 còn điểm ảnh màu trắng mang bit 1, vi điều khiển dựa vào các tín hiệu này để điều khiển van Solenoid đóng mở. Nếu bit 0 thì van mở ngược lại thì đóng. Thời gian đóng mở van xảy ra trong một khoảng thời gian delay rất nhỏ, tạo thành nhiều bọt khí nỗi tự do liên tục. Các điểm bọt khí này khi liên kết với nhau sẽ tạo thành hình ảnh.



Hình 3.3: Ảnh nhị phân chuyển sang mảng 2 chiều

Việc xử lý ảnh này được thực hiện hoàn toàn trên máy tính, máy tính sẽ thực hiện toàn bộ việc xử lý ảnh sau đó mới gửi tín hiệu xuống hệ thống mạch điều khiển van. Vấn đề gặp phải là các van solenoid có điện áp khá lớn (khoảng 24V), vì thế khi hoạt động cùng lúc thì công suất của hệ thống trở nên rất lớn. Do đó không thể điều khiển các van này trực tiếp bằng Vi điều khiển được, mà sẽ được xử lý bởi các mạch Công suất. Các mạch Công suất này sẽ dựa vào tín hiệu từ Vi điều khiển và trực tiếp điều khiển hệ thống van solenoid.



Hình 3.4: Mạch công suất điều khiển van

Một vi điều khiển thông thường có số lượng PORT giới hạn, trong khi đó trong hệ thống Màn hình nước luôn đòi hỏi một số lượng lớn van và phải có khả năng mở rộng thêm sau này. Để giải quyết vấn đề này, hệ thống mạch sử dụng nhiều IC 595 mắc nối tiếp. Với cách này, Màn hình nước có thể mở rộng số lượng van bao nhiêu tuỳ ý.

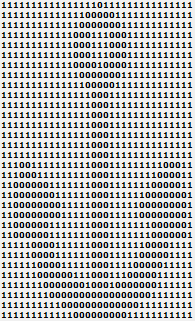
## **3.2. Thiết kế tổng quan hệ thống**

Như vậy qua phân tích các yêu cầu đề ra cho một hệ thống Màn hình nước. Ta có một cái nhìn tổng quan về hệ thống như sau:

Hình ảnh sau xử lý

Hình ảnh

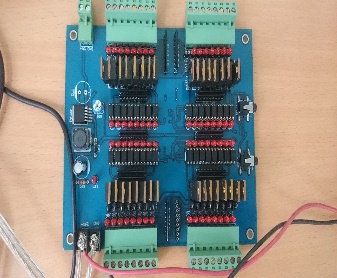
Máy tính xử lý





Gửi tín hiệu xuống mạch điều khiển

Mạch điều khiển van đóng mở

Hình 4: Sơ đồ chi tiết của hệ thống

Như vậy, máy tính sẽ load hình ảnh vào phân tích và sau đó gửi tín hiệu điều khiển cho mạch điều khiển. Mạch điều khiển dựa vào tín hiệu trên sẽ kích vào các mạch Công suất hoạt động, các mạch Công suất sẽ trực tiếp đóng mở các van.

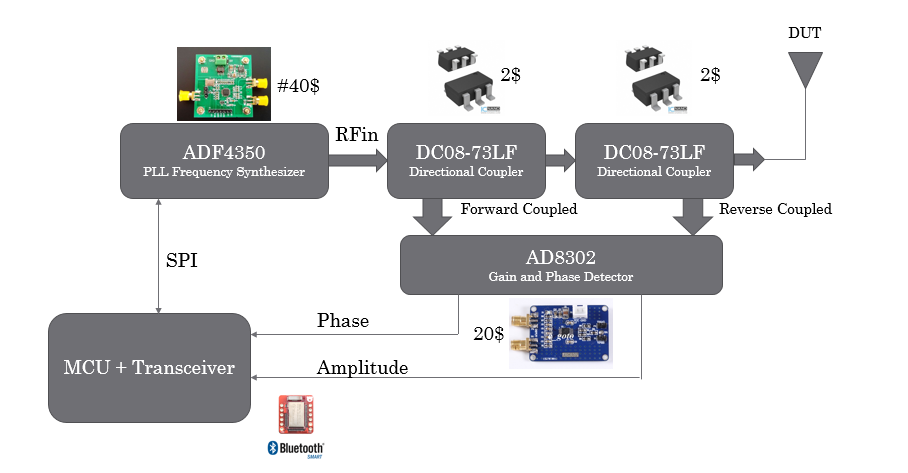
# **CHƯƠNG 4: THIẾT KẾ MẠCH VÀ XÂY DỰNG CHƯƠNG TRÌNH ĐIỀU KHIỂN THIẾT BỊ MINI VECTOR NETWORK ANALYZER**

## **4.1. Thiết kế mạch điều khiển**

### *4.1.1. Sơ đồ khối hệ thống và sơ đồ khối Kit phát triển Renesas GR-Peach*

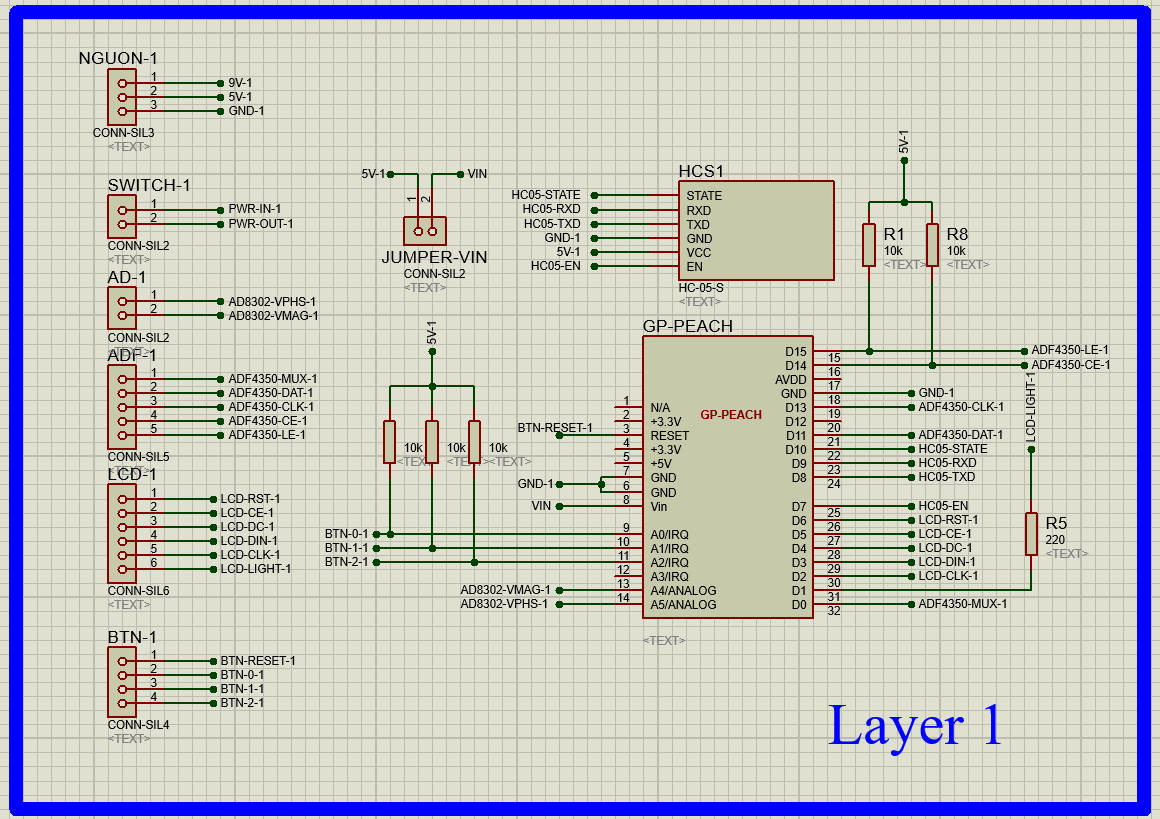
Như đã phân tích ở chương 3, hệ thống mạch đóng vai trò điều khiển trực tiếp thiết bị, hệ thống mạch sẽ nhận tín hiệu gửi từ thiết bị di động, điều khiển các mô-đun hoạt động. Trong hệ thống mạch ta tập trung xây dựng 4 khối chính như sau: Khối điều khiểnt trung tâm (MCU), Khối mạch nguồn LM2596, Khối mô-đun ADF4351 Evaluation Board – AD8302 – Coupler, Khối LCD Nokia 5110 và nút nhấn.

**Sơ đồ khối hệ thống như sau:**



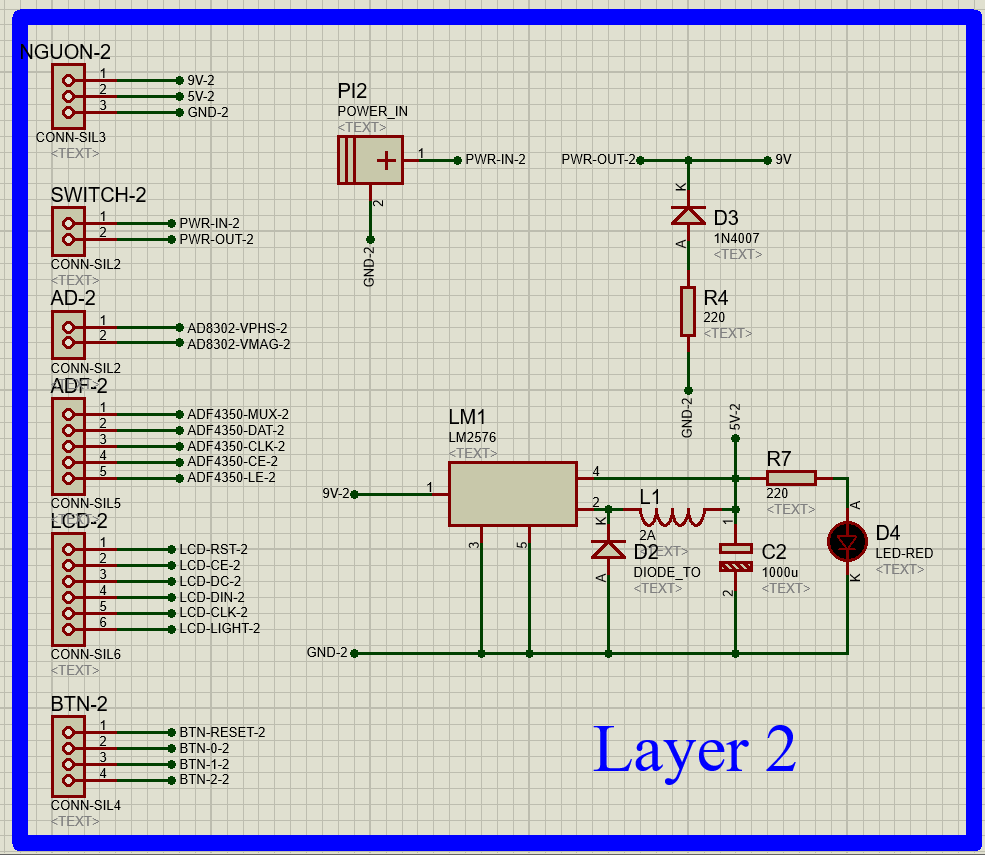
Hình 4.1: Sơ đồ khối Hệ thống mạch điều khiển

**Sơ đồ** **khối điều khiển trung tâm (MCU)**



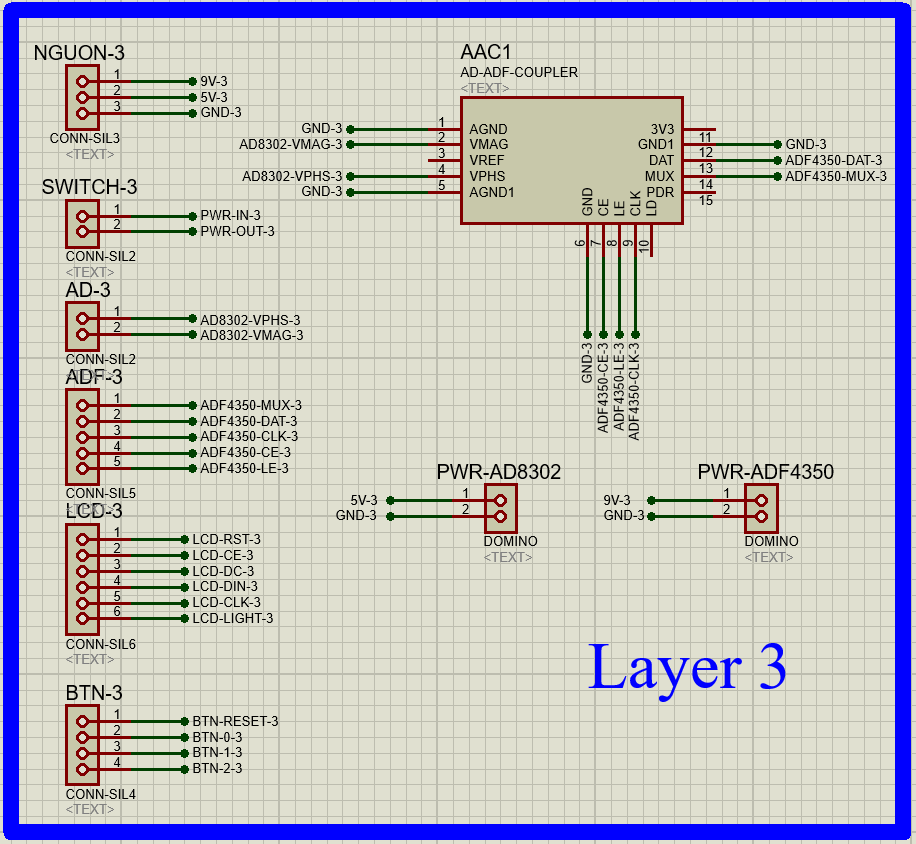
Khối điều khiển trung tâm (MCU) bao gồm kit phát triển Renesas GR-Peach và mô-đun bluetooth HC-05, đóng vai trò là bộ não của thiết bị, chịu trách nhiệm trong việc nhận lệnh từ thiết bị di động, xử lý và điều khiển các khối khác hoạt động theo lệnh đó.

*4.1.2. Khối mạch nguồn LM2596*



Khối này bao gồm mạch nguồn IC LM2596 và các linh kiện điện tử thông dụng, chịu trách nhiệm chuyển đỗi dòng điện từ 9V sang 5V phù hợp cho các linh kiện trong thiết bị như: mô-đun bluetooth HC-05, mô-đun AD8302, LCD Nokia 5110,…

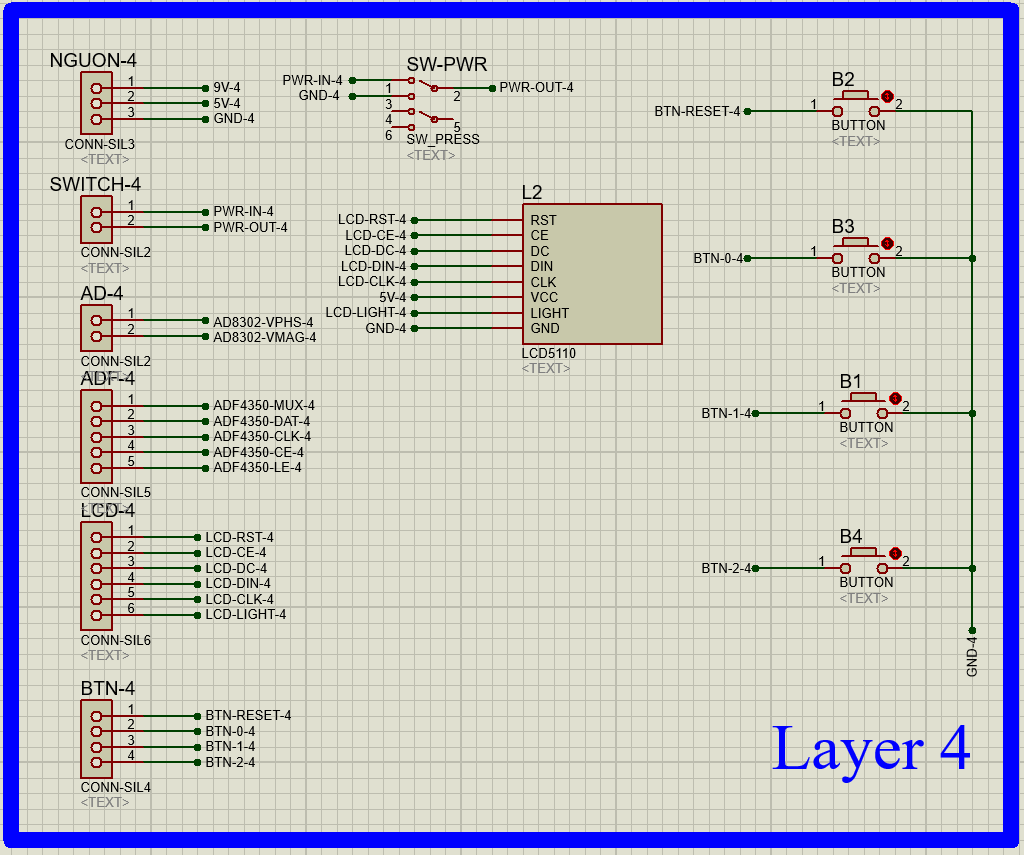
### *4.1.3. Khối mô-đun ADF4351* *Evaluation Board – AD8302 – Coupler*



Khối mô-đun ADF4351 Evaluation Board – AD8302 – Coupler gồm ba thành phần chính là mô-đun ADF4351, mô-đun AD8302 và mô-đun Coupler DC08-73LF.

Sau khi nhận lệnh từ thiết bị di động, khối điều khiển trung tâm điều khiển mô-đun ADF4351 phát ra sóng ở tần số cố định theo yêu cầu của người dùng thông qua giao thức SPI. Sóng này được truyền thẳng vào mô-đun Coupler để tách ra thành hai sóng: sóng tới từ mô-đun ADF và sóng phản xạ về từ antenna. Hai sóng này được đưa vào mô-đun AD8302 để so sánh, đưa ra thông số chênh lệnh của hai sóng này. Từ đó, có thể phân tích thông số và đánh giá được chất lượng của antenna.

*4.1.4. Khối LCD Nokia 5110 và nút nhấn*



Khối LCD Nokia 5110 và nút nhấn sẽ nhận lệnh từ người dùng khi không có thiết bị di động và thể hiện kết quả sau khi đo đạt thành công.

## **4.2. Xây dựng ứng dụng điều khiển và hiển thị kết quả từ thiết bị**

### *4.2.1. Sơ đồ khối*

Ứng dụng Mini Vector Network Analyzer bao gồm 3 màn hình chính:

* Màn hình 1: kết nối với thiết bị.
* Màn hình 2: chọn thông tin tần số cần đo
* Màn hình 3: hiển thị kết quả.

Sơ đồ điều khiển chương trình được thực hiện theo sơ đồ như sau:

Chọn tần số

Nhận kết quả

Hiển thị kết quả

Kết nối

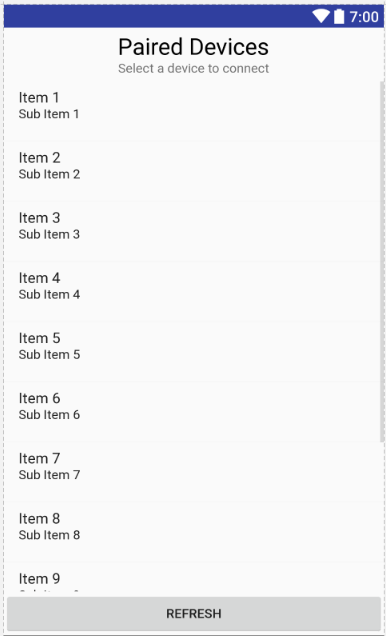
Hình 4.9: Sơ đồ điều khiển chương trình

### *4.2.2.Ứng dụng Mini Vector Network Analyzer*

#### 4.2.2.1. Giới thiệu

Ứng dụng Mini Vector Network Analyzer có nhiệm vụ thực hiện 3 chức năng cơ bản là: kết nối thiết bị di động với VNA, truyền dữ liệu sau khi chọn các tần số cần đo xuống thiết bị, Cuối cùng là hiển thị kết quả đo được.

#### 4.2.2.2. Màn hình 1: kết nối với thiết bị



|  |
| --- |
| <**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"**>   <**LinearLayout  android:id="@+id/linearLayout"  android:orientation="vertical"  android:gravity="center"  android:padding="6dp"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"**>   <**TextView  android:id="@+id/textView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_alignParentLeft="true"  android:layout\_alignParentStart="true"  android:layout\_alignParentTop="true"  android:text="Paired Devices"  android:textSize="25sp"  android:textColor="#000000"**/>   <**TextView  android:text="Select a device to connect"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"** />  </**LinearLayout**>   <**Button  android:id="@+id/btnRefreshList"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_alignParentBottom="true"  android:layout\_alignParentEnd="true"  android:layout\_alignParentLeft="true"  android:layout\_alignParentRight="true"  android:layout\_alignParentStart="true"  android:text="Refresh"** />   <**ListView  android:id="@+id/listView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/linearLayout"  android:layout\_above="@+id/btnRefreshList"** /> </**RelativeLayout**> |

4.2.2.3. Màn hình 2: chọn thông tin tần số cần đo.



|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:gravity="center"  tools:context="com.example.manht.testbluetooth.SetFrequencyActivity"**>   <**TextView  android:id="@+id/tvEnterFrequency"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Enter frequency"  android:textSize="20sp"  android:padding="4dp"  android:textColor="#000000"**/>   <**EditText  android:id="@+id/txtFrequency"  android:layout\_width="match\_parent"  android:layout\_below="@+id/tvEnterFrequency"  android:layout\_height="wrap\_content"  android:padding="16dp"  android:inputType="numberDecimal"  android:digits="0123456789,."  tools:text="315, 433, 868, 2400"** />   <**Button  android:id="@+id/btn315MHz"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequency"  android:text="315 MHz"** />   <**Button  android:id="@+id/btn433MHz"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequency"  android:layout\_toRightOf="@+id/btn315MHz"  android:text="433 MHz"** />   <**Button  android:id="@+id/btn868MHz"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequency"  android:layout\_toRightOf="@+id/btn433MHz"  android:text="868 MHz"** />   <**Button  android:id="@+id/btn2400MHz"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequency"  android:layout\_toRightOf="@+id/btn868MHz"  android:text="2.4 GHz"** />   <**Button  android:id="@+id/btnDel"  android:text="Del"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequency"  android:layout\_toRightOf="@+id/btn2400MHz"**/>   <**Button  android:id="@+id/btnComma"  android:text=","  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/btn315MHz"**/>   <**Button  android:id="@+id/btnSetFrequency"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_alignParentBottom="true"  android:text="Set Frequency"** />  </**RelativeLayout**> |

4.2.2.4. Màn hình 3: hiển thị kết quả.



|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  tools:context="com.example.manht.testbluetooth.DisplayDiagram"**>   <**TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:padding="6dp"  android:text="Measuring Result"  android:textColor="#000000"  android:textSize="24sp"** />   <**TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="left"  android:text="Magnitude (dB)"** />   <**com.github.mikephil.charting.charts.LineChart  android:id="@+id/chart"  android:layout\_width="match\_parent"  android:layout\_height="450dp"**></**com.github.mikephil.charting.charts.LineChart**>   <**TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="right"  android:text="Frequencies (MHz)"** />   <**RelativeLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:gravity="center\_vertical"**>   <**TextView  android:id="@+id/txtFrequencyInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_alignParentTop="true"  android:layout\_alignStart="@+id/txtMagnitudeInfo"  android:layout\_marginTop="24dp"  android:padding="6dp"  android:text="Frequency: "  android:textColor="#000000"  android:textSize="15sp"** />   <**TextView  android:id="@+id/txtShowFrequencyInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_above="@+id/txtMagnitudeInfo"  android:layout\_alignStart="@+id/txtShowMagnitudeInfo"  android:padding="6dp"  android:textColor="#000000"  android:textSize="15sp"  android:text="Not selected"** />   <**TextView  android:id="@+id/txtMagnitudeInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtFrequencyInfo"  android:layout\_marginStart="50dp"  android:padding="6dp"  android:text="Magnitude: "  android:textColor="#000000"  android:textSize="15sp"** />   <**TextView  android:id="@+id/txtShowMagnitudeInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@+id/txtShowFrequencyInfo"  android:layout\_marginStart="11dp"  android:layout\_toEndOf="@+id/txtMagnitudeInfo"  android:padding="6dp"  android:textColor="#000000"  android:textSize="15sp"  android:text="Not selected"** />   <**TextView  android:id="@+id/txtQualityInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_alignStart="@+id/txtMagnitudeInfo"  android:layout\_below="@+id/txtMagnitudeInfo"  android:padding="6dp"  android:text="Quality: "  android:textColor="#000000"  android:textSize="15sp"** />   <**TextView  android:id="@+id/txtShowQualityInfo"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_alignStart="@+id/txtShowMagnitudeInfo"  android:layout\_below="@+id/txtShowMagnitudeInfo"  android:padding="6dp"  android:textColor="#000000"  android:textSize="15sp"  android:text="Not selected"** />   </**RelativeLayout**>  </**LinearLayout**> |

### *4.2.3. Chương trình điều khiển trên kit phát triển Renasas GR-Peach*

#### 4.2.3.1. Giới thiệu

Chương trình điều khiển trên kit phát triển Renasas GR-Peach chỉ đơn giản là nhận dữ liệu từ máy tính gửi xuống qua cổng COM sau đó gửi dữ liệu này cho khối 595 và khối Công suất điều khiển van Solenoid.

#### 4.2.3.2. Xây dựng chương trình điều khiển cho kit phát triển Renasas GR-Peach

Chương trình xử lý trên STM32F4 bao gồm 2 phần: Phần nhận dữ liệu từ máy tính và phần quét 595.

Để nhận dữ liệu từ máy tính cần khai báo một mảng lưu dữ liệu như sau:

*unsigned char rx\_buffer[NUMS]*

Trong đó NUMS là số IC 595 sử dụng trong mạch.

Khi có dữ liệu gửi từ máy tính gửi xuống, Vi điều khiển sẽ thực hiện một ngắt. Ta chỉ việc thực hiện ghi dữ liệu từ thanh ghi vào mảng *rx\_buffer* trong quá trình thực hiện ngắt.

*interrupt [USART\_RXC] void usart\_rx\_isr(void)*

*{*

*if(rx\_index<NUMS){*

*rx\_buffer[rx\_index]=UDR;*

*rx\_index++;*

*}else {*

*rx\_index=0;*

*}*

*}*

Để đưa dữ liệu ra khối 595 sử dụng kĩ thuật quét 595 như sau:

* Trước hết, xây dựng hàm tạo xung cho chân SH (SCK) và ST (LATCH) của IC 595:

*//Tao xung nhip chan SCK de ghi DATA vao 595*

*void sck(){*

*SCK=1;*

*SCK=0;*

*}*

*//Tao xung nhip chan LATCH de xuat du lieu tu 595*

*void latch(){*

*LATCH=1;*

*LATCH=0;*

*}*

* Thực hiện quét từng byte vào mạch 595:

*//Quet 595*

*void Quet(unsigned char x){*

*unsigned char temp;*

*for(n=0;n<8;n++)*

*{*

*temp=x;*

*temp=temp&0x80;*

*if(temp==0x80){*

*DATA=1;*

*}else {DATA=0;}*

*sck();*

*x=x<<1;*

*}*

*}*

*//DATA là chân DS của IC 595.*

* Kết hợp quét dữ liệu lưu trong *rx\_buffer* và điều khiển mạch 595 kích điều khiển van Solenoid như sau:

*void Display(){*

*if(rx\_index==NUMS){*

*for(k=NUMS;k>0;k--){*

*Quet(rx\_buffer[k-1]);*

*}*

*latch();*

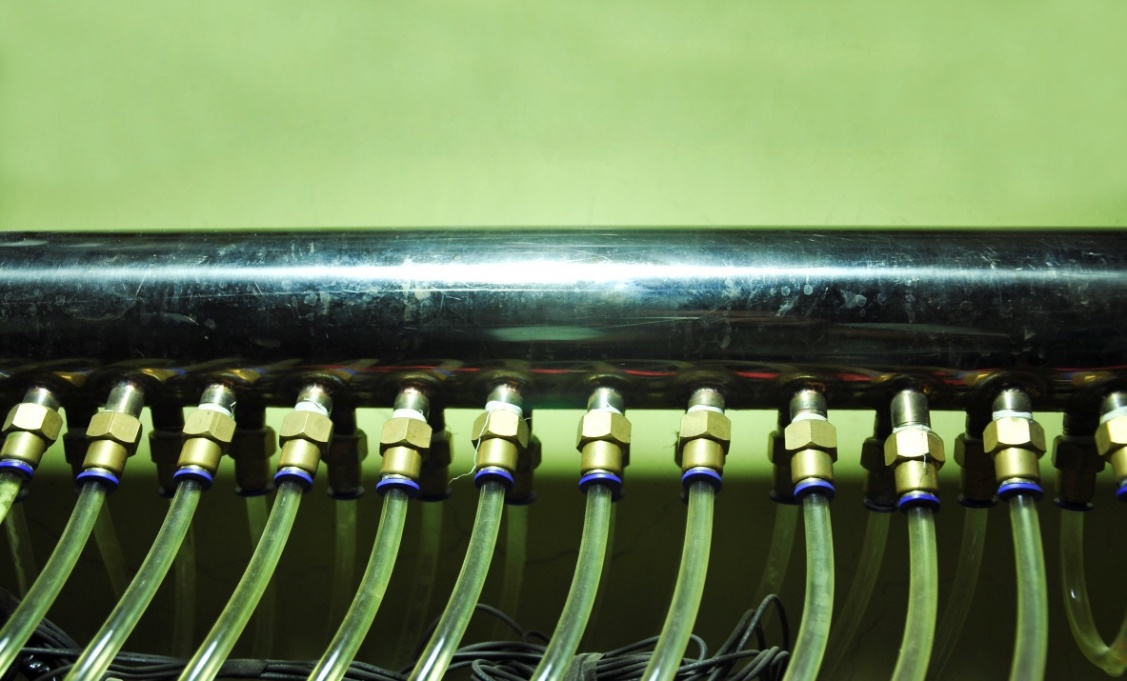
*}*

*}*

Như vậy, chương trình điều khiển cho STM32F4 cơ bản đã được xây dựng xong. Chương trình này thực hiện việc nhận dữ liệu từ máy tính qua cổng COM ghi vào một mảng rồi từ mảng thực hiện quét ra IC 595.

## **4.3. Thiết kế lắp đặt dàn Solenoid**

Bubble Graphic Display sử dụng van Solenoid của hãng STNC. Van này có điện áp là 24V-50Mhz, 8W. Các van này lắp đặt liên tiếp nhau thành một hàng ngang. Phía trên là ống chứ nước đặt sát nhau.

**

Hình 4.16: Ống chứa khí bơm đẩy lên các đầu van.



Hình 4.17: Khung.

Màn hình nước được treo cao hơn mặt đất khoảng 3m có bơm nước bơm liên tục vào ống Inox. Dây điện từ các van được nối trực tiếp vào các mạch Công suất của hệ thống điều khiển.

## **4.4. Kết quả thực nghiệm thu được**

### *4.4.1. Hệ thống mạch điều khiển*

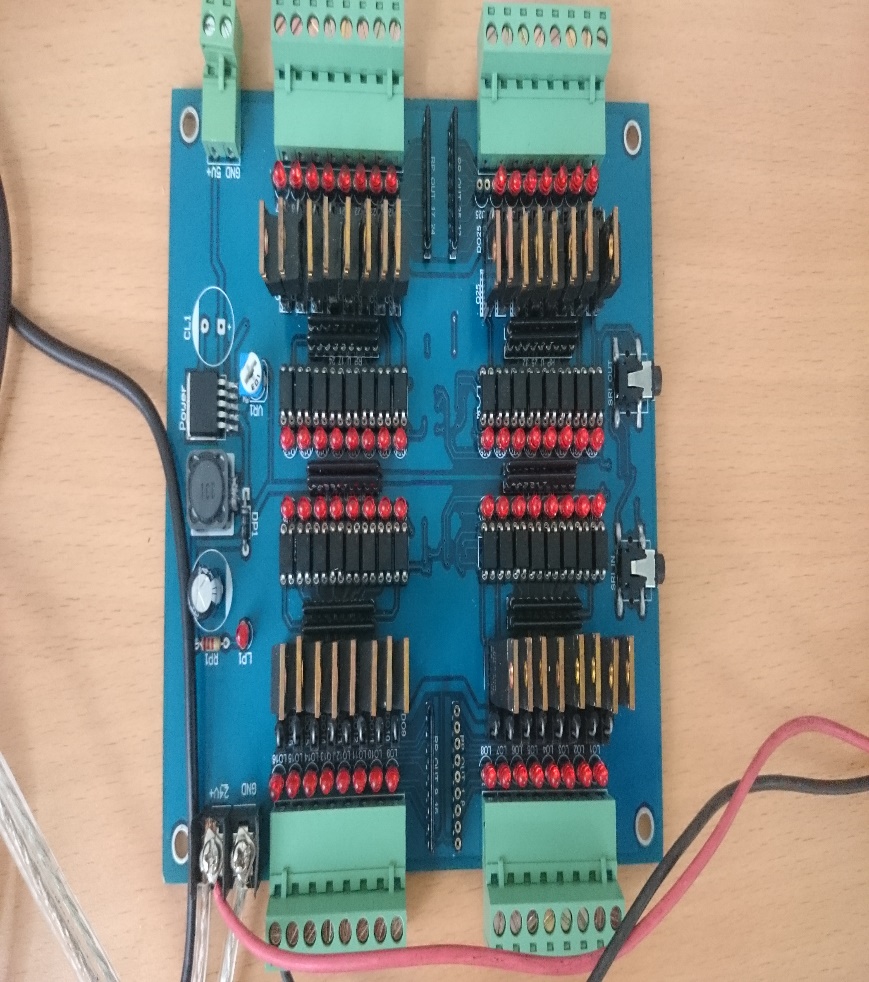
Sau khi thiết kế, chạy layout và làm mạch, hàn linh kiện, kết quả thu được như sau:

* Khối mạch mạch điều khiển trung tâm (MCU):



Hình 4.18: Mạch Vi điều khiển

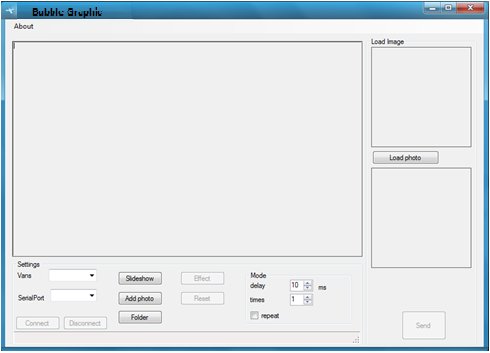
* Khối mạch nguồn LM2596:



Hình 4.19: Mạch Công suất

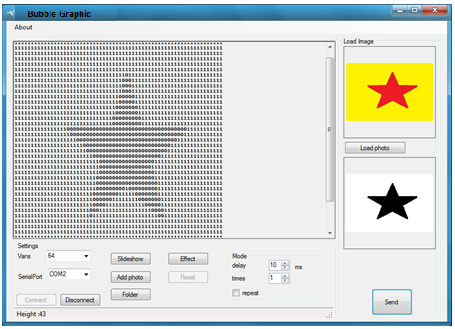
### *4.4.2. Ứng dụng Mini Vector Network Analyzer*

### Ứng dụng Mini Vector Network Analyzer có giao diện chương trình như sau:



Hình 4.20: Giao diện Ứng dụng Mini Vector Network Analyzer

Load ảnh vào chương trình. Chương trình chuyển ảnh màu sang ảnh trắng đen và mảng 2 chiều.



Hình 4.21: Giao diện chương trình khi load hình, thiết lập thông số

Mảng này sẽ được gửi xuống mạch điều khiển qua cổng COM 2.

Chú thích các chức năng của chương trình:

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên** | **Chức năng** |
| 1 | Vans | Chọn số Van Solenoid có trên hệ thống. |
| 2 | SerialPort | Chọn tên cổng COM của máy. |
| 3 | Connect | Kết nối cổng COM với mạch điều khiển. |
| 4 | Disconnect | Đóng kết nối. |
| 5 | Delay | Chọn thời gian delay. |
| 6 | Times | Chọn số lần rớt ảnh. |
| 7 | Repeat | Lặp đi lặp lại chương trình rơi ảnh. |
| 8 | Slideshow | Rơi danh sách ảnh do người dùng chọn từ trước |
| 9 | Effect | Rơi 1 số hiệu ứng/ ảnh đã tích hợp trong chương trình |
| 10 | Add Photo | Thêm ảnh vào danh sách ảnh hiển thị |
| 11 | Load Photo | Chọn 1 ảnh vào chương trình |
| 12 | Send | Gửi mảng sau xử lý xuống mạch điều khiển |
| 13 | Folder | Chọn thư mục chứa các ảnh cần hiển thị |

Bảng 4.3: Liệt kê các chức năng của chương trình phần mềm

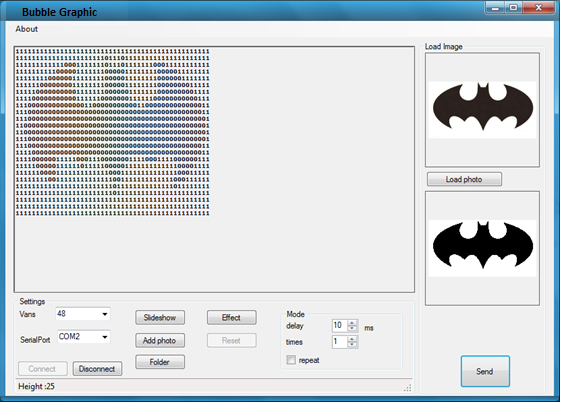
### *4.4.3. Kết quả thực nghiệm*

**Chuẩn bị ứng dụng:**

**B1**

Chạy ứng dụng trên thiết bị di động

Chọn kênh Bluetooth của thiết bị và kết nối



Hình 4.22: Xử lý 1 ảnh trên phần mềm và gửi xuống mạch điều khiển

B2

Chọn các tần số tương ứng cần cho việc đo đạt

B3

Chờ và xem kết quả được gửi về từ thiết bị

**Chuẩn bị phần cứng:**

Cung cấp nguồn 9V vào mạch điều khiển, kết nối các khối mạch với nhau. Kết quả thu được:



Hình 4.23: Máy bơm khí mini



Hình 4.24: Hình ảnh sản phẩm thực tế

## **4.5. Đánh giá**

**Ưu điểm:** Mạch chạy ổn định, chương trình xử lý khá chính xác.

**Khuyết điểm:** Do sử dụng màn hình LCD có kích thước nhỏ ( 1,6 inches) nên khi hoạt động độc lập không có thiết bị di động thì chỉ có thể đo được ở một số tần cố định thông dụng ( 315 Mhz, 433 Mhz, 868 Mhz, 2.4 Ghz).

**CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **5.1. Kết luận**

Sau khi hoàn thành đề tài này, về cơ bản đã xây dựng được một thiết bị phân tích, đánh giá chất lượng antenna trong các thiết bị IoT nhỏ, đơn giản có thể đo và đánh giá antenna ở một số tần số thông dụng hay một dãy tần số (khi có thiết bị di động) tương đối chính xác. Mặc dù chưa thể so sánh được với các thiết bị chuyên dụng về độ chính xác, song đó là bước khởi đầu để phát triển một thiết bị phân tích, đánh giá chất lượng antenna giá thành thấp, nhỏ gọn. Dưới đây là những kết quả mà đề tài đạt được.

* Về mặt lý thuyết:
  + Nắm được nguyên lý đánh giá chất lượng antenna
  + Hiểu biết thêm về nguyên lý hoạt động của các thiết bị đo, đánh giá sóng chuyên dụng (SNA, VNA, …)
  + Tìm hiểu thêm một số kĩ thuật cơ bản trong truyền nhận dữ liệu qua Bluetooth
* Về mặt thực nghiệm:
  + Thiết kế hệ thống mạch nhiều tầng, chiếm diện tích nhỏ
  + Xây dựng chương trình ứng dụng điều khiển thiết bị trên hệ điều hành Android. Cho phép điều khiển thiết bị từ xa và hiển thị kết quả một cách trực quan.
  + Cho chạy thử nghiệm thiết bị và đạt được một số kết quả khả quan khi so sánh với các thiết bị chuyên dụng.

## **5.2. Hướng phát triển**

Bên cạnh các kết quả đạt được, đề tài vẫn còn gặp nhiều hạn chế cần phải khắc phục thêm. Đó cũng chính là hướng nghiên cứu mở rộng trong tương lai:

* + Nghiên cứu giao tiếp giữa thiết bị với Internet để người dùng có thể điều khiển từ xa, cụ thể là thông qua hệ thống website.
  + Nghiên cứu tích hợp các mô-đun rời rạc thành một mô-đun nhỏ gọn hơn, thay vì dùng các mô-đun thiết kế sẵn trên thị trường.
  + Cải thiện giao diện của ứng dụng điều khiển thiết bị trên điện thoại.
  + Phát triển và mở rộng ứng dụng điều khiển trên các hệ điều hành khác như: IOS, Windows Phone,…
  + Nâng cấp khả năng hoạt động độc lập cho thiết bị, cụ thể là có thể đo một dãy tầng số và hiển thị đồ thị kết quả trên chính thiết bị.
  + Cải thiện độ chính xác cũng như giảm giá thành.

# **TÀI LIỆU THAM KHẢO**

* **The ARRL Antenna Book** (21st ed.), The American Radio Relay League, Inc., 2007, ISBN 0-87259-987-6
* **Introduction to Network Analyzer Measurements Fundamentals and Background** - National Instruments, http://www.ni.com/rf-academy
* Johannes Nehring, Marco Dietz, Robert Weigel, **Highly Integrated 4–32-GHz Two-Port Vector Network Analyzers for Instrumentation and Biomedical Applications**