

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA KỸ THUẬT ĐIỆN TỬ I**

-----🙞🙜🕮🙞🙜-----

**BÁO CÁO BÀI TẬP LỚN**

*Đề tài:*

*Nghiên cứu và thử nghiệm mạch cảm biến hồng ngoại*

***Giảng viên hướng dẫn: ThS. Trương Minh Đức***

***Sinh viên thực hiện: Bùi Thiện Anh***

***Mã SV: B21DCDT031***

***Lớp: D21CQDT03-B***

***Hệ đào tạo: Chính quy***

**Hà Nội, 2024**

# **LỜI CẢM ƠN**

Mặc dù chỉ là một môn học 2 tín chỉ nhưng đây là những viên gạch đầu tiên để đặt nền móng cho định hướng tương lai của em sau này. Và để đạt được kết quả như hiện tại thì không thể nào không nhắc tới những sự hỗ trợ trực tiếp lẫn gián tiếp của các giảng viên, bạn bè xung quanh.

Lời đầu tiên em xin được bày tỏ sự biết ơn sâu sắc đến Ban giám đốc Học viện Công nghệ Bưu chính Viễn thông đã taọ cho em môi trường học tập, rèn luyện, trau dồi kiến thức và kinh nghiệm.

Cảm ơn những người bạn trong và ngoài nhóm đã cùng đồng hành, động viên, cổ vũ và giúp đỡ tôi trực tiếp lẫn gián tiếp trong quá trình thực hiện đề tài từ khâu lên ý tưởng đến khi hoàn thiện sản phẩm vật lý.

Đặc biệt, em xin bày tỏ lòng biết ơn chân thành sâu sắc tới thầy **Trương Minh Đức** người trực tiếp hướng dẫn và cung cấp cho em những kiến thức, tài liệu quý giá và giúp em định hướng cho các bước tìm hiểu nghiên cứu và mô phỏng để em hoàn thành tốt đề tài này.

Con xin gửi lời cảm ơn đến cha mẹ, những người luôn động viên và khích lệ, tạo mọi điều kiện thuận lợi nhất trong quá trình học tập và thực hiện đề tài này của con.

Hà Nội, ngày tháng năm 2024

**Sinh viên thực hiện**

Bùi Thiện Anh

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_Toc2617)

[DANH MỤC HÌNH ẢNH 3](#_Toc3288)

[LỜI NÓI ĐẦU 4](#_Toc15446)

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 5](#_Toc22647)

[1. Lý do chọn đề tài 5](#_Toc21045)

[2. Ưu điểm của thiết bị 5](#_Toc20683)

[3. Nhược điểm của thiết bị 5](#_Toc20231)

[4. Mục tiêu đề ra 6](#_Toc28170)

[5. Kết luận chương 6](#_Toc27857)

[CHƯƠNG 2: GIỚI THIỆU CÁC LINH KIỆN ĐƯỢC SỬ DỤNG 7](#_Toc8764)

[1. Kit phát triển STM32F103C8T6 Blue Pill 7](#_Toc23847)

[2. Modun IR LM393 9](#_Toc2451)

[3. Biến Trở Vi Chỉnh 3296W 11](#_Toc24761)

[4. Mạch nạp ST-Link V2 Mini 12](#_Toc10807)

[5. Còi chíp 5V TMB12A05 12x9.5mm 12](#_Toc29794)

[6. Kết luận chương 13](#_Toc10144)

[CHƯƠNG 3: LẬP TRÌNH CHO VI ĐIỀU KHIỂN 14](#_Toc1945)

[1. Thuật toán cảm biến vật cản 14](#_Toc11716)

[2. Lập trình bằng cho vi điều khiển bằng phần mềm STM32CubeIDE 14](#_Toc3952)

[3. Kết luận chương 25](#_Toc15001)

[CHƯƠNG 4: THIẾT KẾ PCB 26](#_Toc8837)

[1. Sơ đồ nguyên lý 26](#_Toc24763)

[2. Layout của PCB 27](#_Toc28055)

# **DANH MỤC HÌNH ẢNH**

[Hình 1: Vi điều khiển STM32S103F3P6 7](#_heading=h.3rdcrjn)

[Hình 2: AMS1117-3.3V 8](#_heading=h.lnxbz9)

[Hình 3: Module cảm biến khí gas MQ2 9](#_heading=h.1ksv4uv)

[Hình 4: Mạch nạp ST-Link V2 Mini 10](#_heading=h.2jxsxqh)

[Hình 5: Lưu đồ thuật toán 11](#_heading=h.2xcytpi)

[Hình 6: Sơ đồ nguyên lý 15](#_heading=h.3as4poj)

[Hình 7: Sơ đồ khối 16](#_heading=h.1pxezwc)

[Hình 8: Width 17](#_heading=h.2p2csry)

[Hình 9: RoutingVias 17](#_heading=h.147n2zr)

[Hình 10: Clearance 18](#_heading=h.3o7alnk)

[Hình 11: Đi dây trên Top Layer 18](#_heading=h.23ckvvd)

[Hình 12: Đi dây trên Bottom Layer 19](#_heading=h.ihv636)

[Hình 13: Mạch sau khi đổ đồng 19](#_heading=h.32hioqz)

[Hình 14: Mô phỏng 3D của mạch 20](#_heading=h.1hmsyys)

[Hình 15: Sản phẩm thực tế 20](#_heading=h.41mghml)

# **LỜI NÓI ĐẦU**

Trong bối cảnh ngày nay, an toàn về phòng cháy chữa cháy trở thành một vấn đề quan trọng không chỉ trong các ngành công nghiệp mà còn trong các môi trường dân dụng. Sự rò rỉ khí gas có thể gây ra nguy hiểm đối với sức khỏe và tính mạng của con người cũng như gây ra các thiệt hại về tài sản.

Đồ án này tập trung vào việc phát triển một hệ thống cảm biến rò rỉ khí gas, một công cụ quan trọng trong việc phát hiện và ngăn chặn sự cố rò rỉ khí gas. Hệ thống sẽ sử dụng cảm biến chuyên dụng để phát hiện có mặt của khí gas trong môi trường và cảnh báo người dùng khi phát hiện rò rỉ.

Đồ án mặc dù khá cơ bản và đơn giản nhưng là những bước đầu tạo tiền đề làm quen với các vi điều khiển, phần mềm nhúng và thiết kế PCB.

Với những kiến thức em có được và tìm hiểu từ bên ngoài, em xin đề xuất làm đồ án với đề tài: “**Hệ thống cảnh báo rò rỉ khí gas sử dụng STM8S103F3P6**” vào trong việc phát hiện và cảnh báo rò rỉ khí gas. Trong đồ án này của em được chia làm 4 chương chính:

**Chương 1:** Tổng quan về đề tài

**Chương 2:** Giới thiệu các linh kiện được sử dụng

**Chương 3:** Lập trình cho vi điều khiển

**Chương 4:** Thiết kế PCB

Bước đầu đi vào thực tế, mặc dù đã có nhiều cố gắng nhưng việc tìm hiểu nghiên cứu cũng như kiến thức của em còn hạn chế và bỡ ngỡ. Do đó chắc chắn đề tài có rất nhiều thiếu sót, em rất mong nhận được sự chỉ bảo, góp ý của Quý Thầy Cô cùng các bạn. Dưới đây em xin trình bày chi tiết các phần trong nội dung của đồ án:

# **CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI**

## **1. Lý do chọn đề tài**

- Hệ thống cảnh báo rò rỉ khí gas là một hệ thống đơn giản vừa đáp ứng được yêu cầu về làm quen, tìm hiểu về các phần mềm nhúng, phần mềm thiết kế PCB, các loại linh kiện và cách lắp ráp các loại linh kiện này. Hệ thống bao gồm cảm biến để đo khí gas trong mỗi trường và đèn led, loa để cảnh báo.

- Lý do lựa chọn vi điều khiển STM8S103F3P6:

* STM8S103F3P6 là một trong những vi điều khiển 8 bit có giá thành rẻ nhất trên thị trường nhưng vẫn cung cấp một loạt các tính năng và chức năng cần thiết cho hệ thống nhúng. Điều này giúp tiết kiệm chi phí sản xuất nhưng vẫn đáp ứng tốt những yêu cầu của thiết bị.
* STM8S103F3P6 được sản xuất bởi STMicroelectronics, một trong những nhà sản xuất hàng đầu trong lĩnh vực vi điều khiển. Sản phẩm của họ thường được đánh giá cao về độ tin cậy và hiệu suất.
* STM8 là dòng vi điều khiển phổ biến, có sẵn nhiều tài liệu tham khảo, hướng dẫn và được hỗ trợ trên nhiều phần mềm lập trình phổ biến như Arduino, IAR,…
* STM8S103F3P6 có sẵn kit phát triển trên thị trường, thuận lợi cho việc chạy mô phỏng hệ thống.

Tóm lại, vi điều khiển STM8S103F3P6 là một lựa chọn hợp lý cho một hệ thống cảnh báo rò rỉ khí gas đơn giản nhờ vào hiệu suất, giá thành, dễ sử dụng, tính tin cậy và số lượng tài liệu tham khảo có sẵn.

## **2. Ưu điểm của thiết bị**

* Sử dụng những linh kiện phổ biến với giá thành rẻ, giúp tiết kiệm chi phí và dễ dàng sửa chữa thay thế.
* Sử dụng nguồn điện 5V từ cổng micro usb nên có thể tận dụng các loại sạc điện thoại có sẵn làm nguồn.
* Cơ chế hoạt động đơn giản, hiệu quả, chỉ sử dụng giao thức GPIO nên dễ dàng cho việc lập trình.

## **3. Nhược điểm của thiết bị**

* Thiết bị đơn giản nên không có nhiều tính năng nâng cao.
* Không giống như kit, vi mạch được thiết kế với chức năng cố định nên không thể sửa đổi bổ sung các tính năng khác.

## **4. Mục tiêu đề ra**

* Làm quen với phần mềm với phần mềm IAR, nắm được cơ bản các tính năng GPIO và TIMER đẻ phục vụ cho việc lập trình.
* Làm quen với phần mềm Altium Designer, thành thạo các kĩ năng cơ bản khi thiết kế mạch như: cài đặt thư viện, vẽ schematic, lựa chọn và sắp xếp linh kiện, thiết lập các rule, đi dây,…
* Tự hoàn thiện, lắp ráp các linh kiện cắm và linh kiện dán kích thước nhỏ.

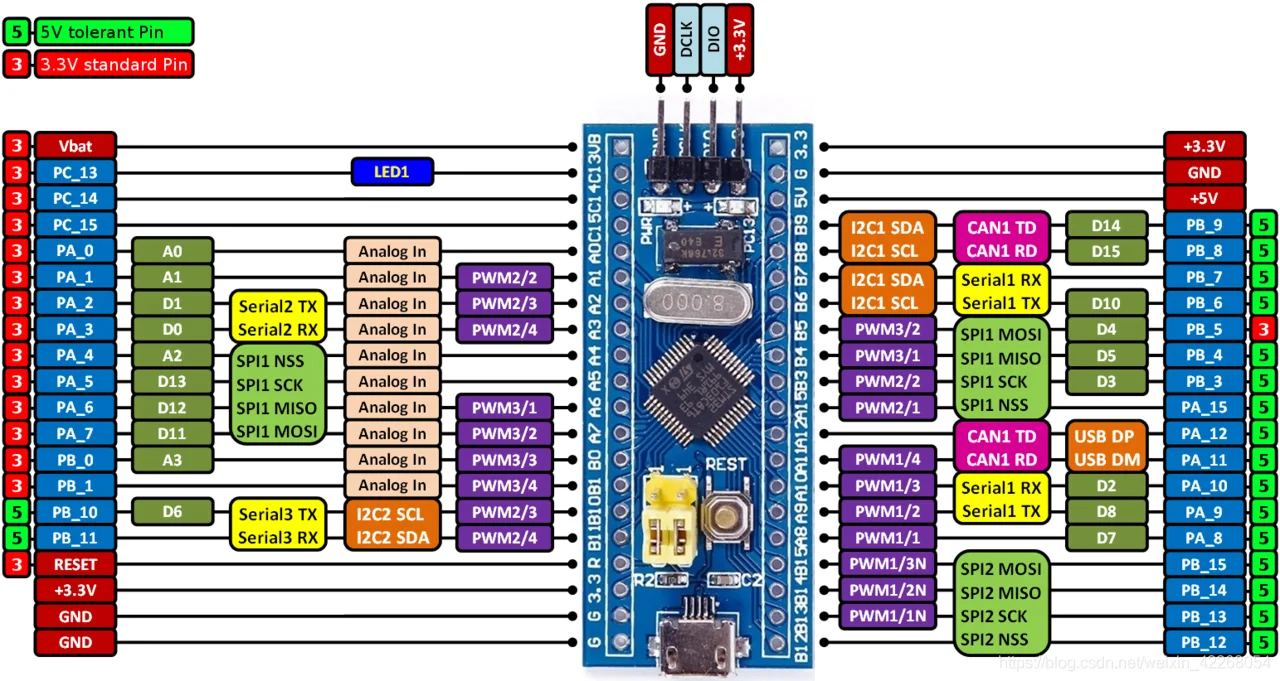
## **5. Kết luận chương**

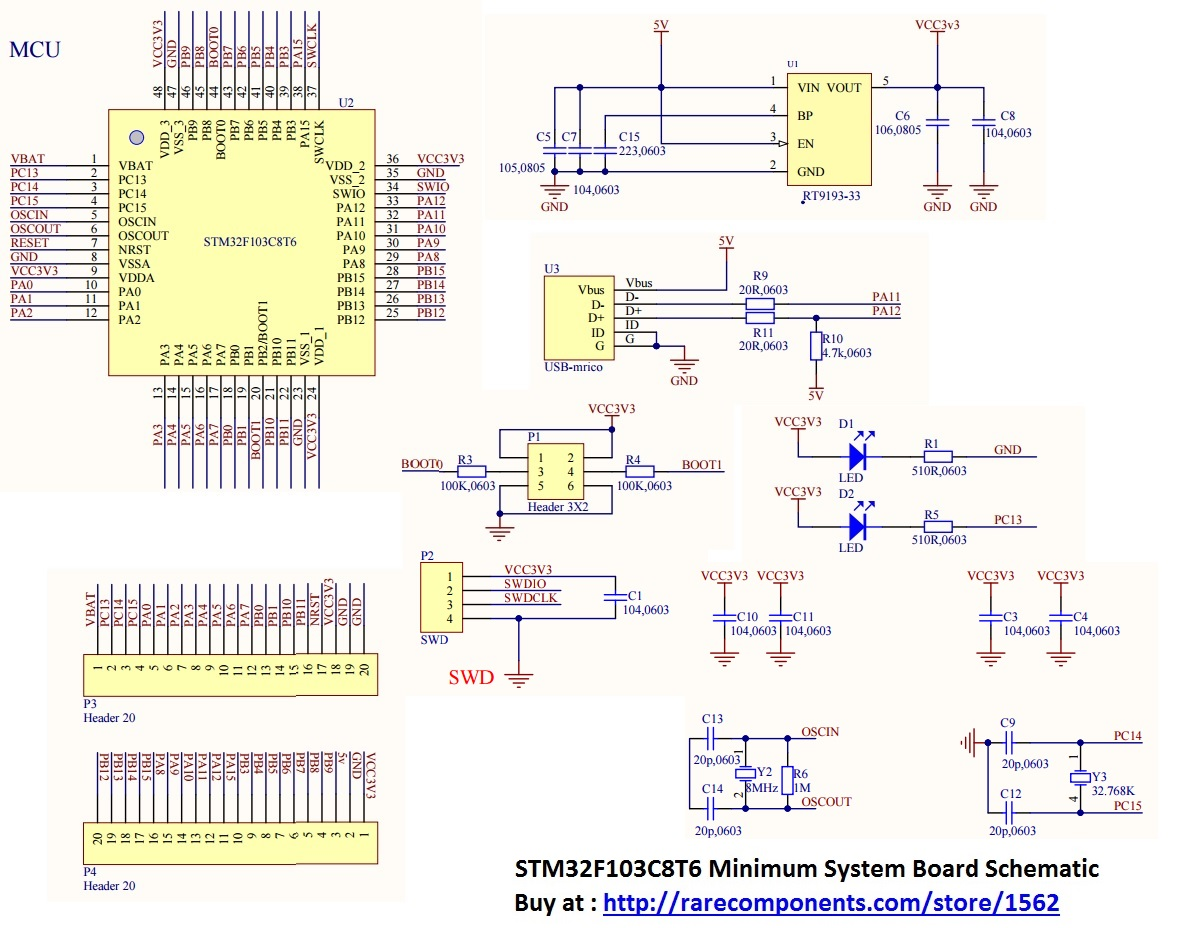
- Ở chương 1 Tổng quan về đề tài em đã trình bày những thông tin cơ bản nhất về mô hình sản phẩm và một ưu nhược điểm của thiết bị cũng như mục tiêu hướng tới của nhóm. Sang chương 2 em sẽ giới thiệu về các linh kiện được sử dụng trong đồ án.

# **CHƯƠNG 2: GIỚI THIỆU CÁC LINH KIỆN ĐƯỢC SỬ DỤNG**

## **Kit phát triển STM32F103C8T6 Blue Pill**

- Kit phát triển STM32F103C8T6 là một nền tảng phần cứng giá rẻ và mạnh mẽ dựa trên vi điều khiển STM32F103C8T6 của STMicroelectronics. Được đặt tên theo hình dạng và màu sắc của nó, Blue Pill là một lựa chọn phổ biến cho các nhà phát triển, học sinh và người đam mê điện tử để học tập và thực hiện các dự án nhúng.





- Thông số:

+ Vi điều khiển: STM32F103C8T6.

+ Điện áp cấp 5VDC qua cổng Micro USB sẽ được chuyển đổi thành 3.3VDC qua IC nguồn và cấp cho Vi điều khiển chính.

+ Tích hợp sẵn thạch anh 8Mhz.

+ Tích hợp sẵn thạnh anh 32Khz cho các ứng dụng RTC.

+ Ra chân đầy đủ tất cả các GPIO và giao tiếp: CAN, I2C, SPI, UART, USB,...

+ Tích hợp Led trạng thái nguồn, Led PC13, Nút Reset.

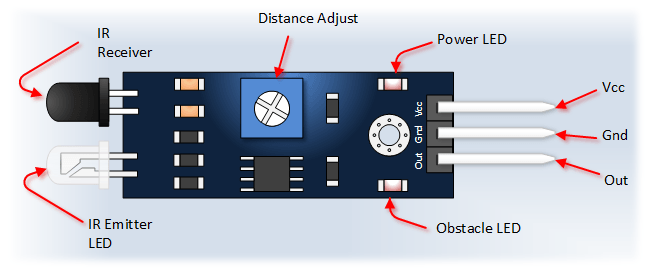
+ Kích thước: 53.34 x 15.24mm.

## **2. Modun IR LM393**

**-** Cảm biến có khả năng nhận biết vật cản ở môi trường với một cặp LED thu phát hồng ngoại để truyền và nhận dữ liệu hồng ngoại. Tia hồng ngoại phát ra với tần số nhất định, khi có vật cản trên đường truyền của LED phát nó sẽ phản xạ vào LED thu hồng ngoại, khi đó LED báo vật cản trên module sẽ sáng, khi không có vật cản, LED sẽ tắt.

Với khả năng phát hiện vật cản trong khoảng 2 ~ 30cm và khoảng cách này có thể điều chỉnh thông qua chiết áp trên cảm biến cho thích hợp với từng ứng dụng cụ thể như: xe dò line, xe tránh vật cản, …





Thông số kỹ thuật :

- IC so sánh: LM393

- Điện áp: 3.3V - 6VDC

- Dòng tiêu thụ:

+ Vcc = 3.3V: 23 mA

+ Vcc = 5.0V: 43 mA

- Góc hoạt động: 35°

- Khoảng cách phát hiện: 2 ~ 30 cm

- LED báo nguồn và LED báo tín hiệu ngõ ra

- Mức logic ngõ ra:

+ Mức thấp - 0V: khi có vật cản

+ Mức cao - 5V: khi không có vật cản

- Kích thước: 3.2cm x 1.4cm

Pinout

Ký hiệu pin Mô tả

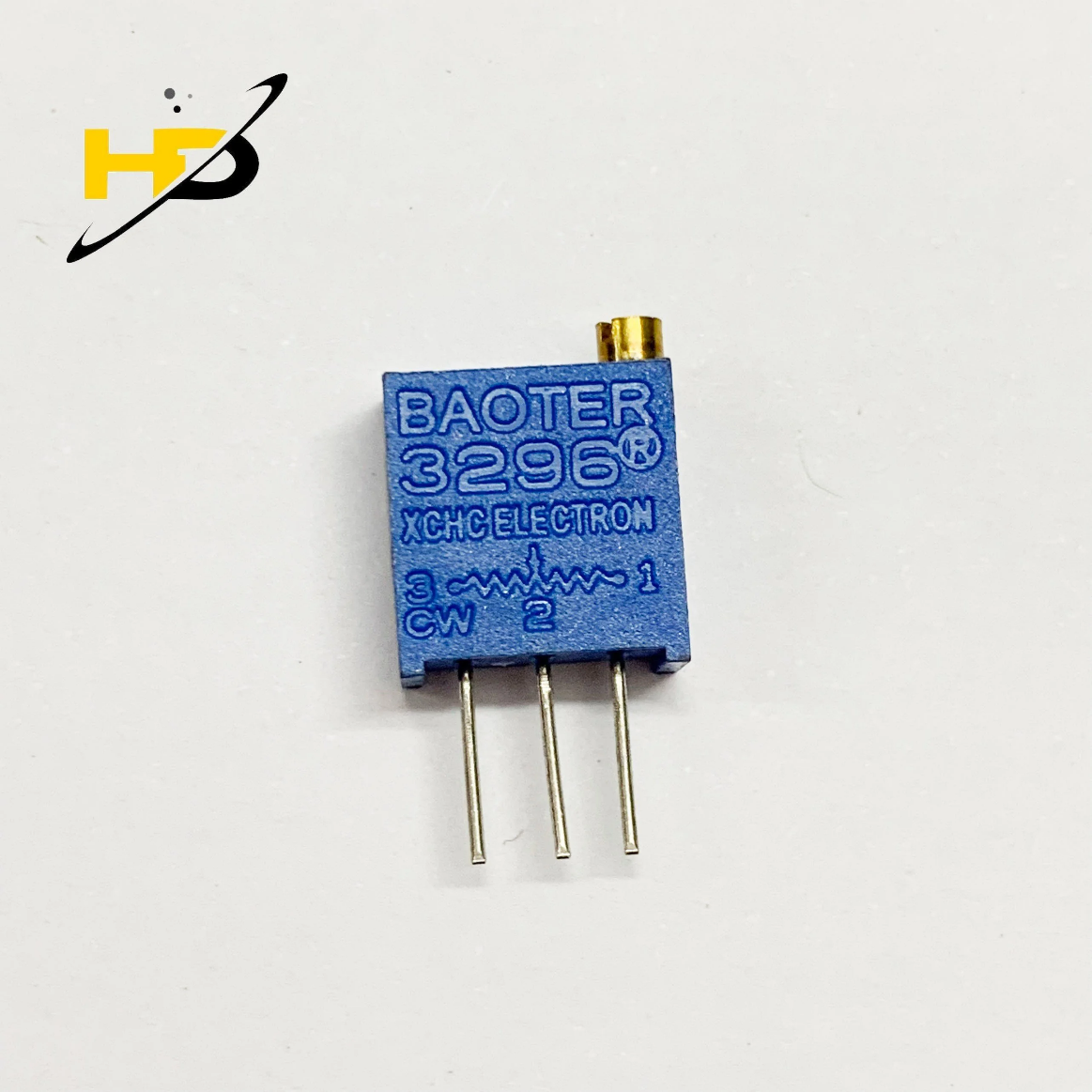
+ VCC Pin nguồn 3.3V/5V

+ GND 0V

+ OUT (digital) Ngõ ra số

## **3. Biến Trở Vi Chỉnh 3296W**

Biến trở tinh chỉnh 3296 dùng để lắp ráp theo mạch điện tử điều chỉnh điện trở chính xác, ví dụ như điều chỉnh độ tinh chỉnh của màn hình LCD 16x2...; Điều chỉnh tốc độ động cơ chạy nhanh hay chậm. Nút điều chỉnh ở phía trên biến trở có thể xoay được để điều chỉnh giá trị điện trở theo ý muốn.



- Thông số:

+ Model: 3296W

+ Số chân biến trở: 3 chân

+ Sai số: 10%

+ Điện áp tối đa: 640VAC

+ Nhiệt độ hoạt động -55°C-125°C

## **4. Mạch nạp ST-Link V2 Mini**

- Mạch nạp STM8, STM32 ST-Link V2 được sử dụng để nạp chương trình và debug cho Vi điều khiển STM32 và STM8 của ST, mạch nạp có kích thước nhỏ gọn, chi phí thấp, độ bền cao.

- Các phần mềm hỗ trợ: STM32CubeProgrammer, ST Visual Programmer, Keil µVision, IAR Embedded Workbench,…



*Hình 4: Mạch nạp ST-Link V2 Mini*

## **5. Còi chíp 5V TMB12A05 12x9.5mm**

Còi chíp TMB12A05 là một thiết bị phát âm thanh điện tử nhỏ gọn, hoạt động ở điện áp 5V. Với kích thước chỉ 12x9.5mm, nó được sử dụng rộng rãi trong các hệ thống điện tử để cung cấp cảnh báo hoặc báo hiệu âm thanh.



Đặc Điểm Kỹ Thuật:

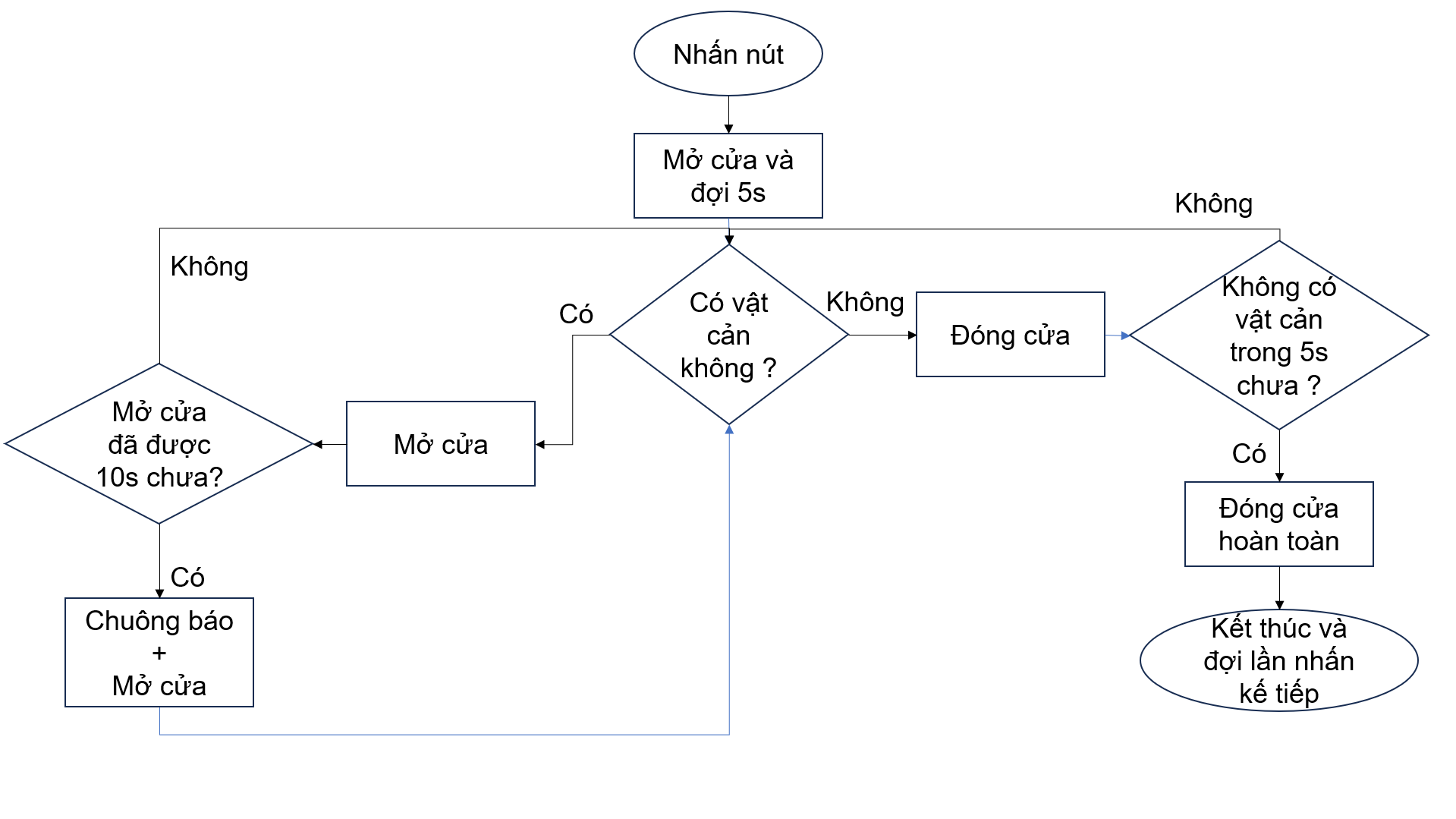
* Điện Áp Hoạt Động: 5V DC
* Dòng Điện Tiêu Thụ: < 30mA
* Tần Số Âm Thanh: 2300 ± 300 Hz
* Cường Độ Âm Thanh: > 85 dB (ở khoảng cách 10cm)
* Kích Thước: 12mm x 9.5mm
* Nhiệt Độ Hoạt Động: -20°C đến +70°C

## **6. Kết luận chương**

- Ở chương 2 này em đã trình bầy khái quát qua về các linh kiện cần thiết sử dụng trong đồ án cũng như các thông số kỹ thuật, nguyên lý hoạt động của các linh kiện. Sang chương tiếp theo em sẽ trình bày về quá trình lập trình và nạp chương trình cho vi điều khiển.

# **CHƯƠNG 3: LẬP TRÌNH CHO VI ĐIỀU KHIỂN**

## **1. Thuật toán cảm biến vật cản**



*Hình 5: Lưu đồ thuật toán*

## **Lập trình bằng cho vi điều khiển bằng phần mềm STM32CubeIDE**

Source code:

#include "main.h"

#include "LCD1602.h"

TIM\_HandleTypeDef htim1;

void SystemClock\_Config(void);

static void MX\_GPIO\_Init(void);

static void MX\_TIM1\_Init(void);

int IR\_level; // TIN HIEU TU CAM BIEN

int row=0; // CHI SO COT TREN LCD

int col=0; // CHI SO HANG TREN LCD

int khoitao=1; // TIN HIEU MO CUA BANG NUT NHAN

int demdong=0; // DEM THOI GIAN ĐỂ DONG CUA

int tinhieu=1; // DUNG DE KHONG THUC THI LENH KHI DA DONG CUA

int demmo=0; // MO LAU QUA THI LOA KEU

int main(void)

{

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

MX\_TIM1\_Init();

HAL\_TIM\_Base\_Start(&htim1);

lcd\_init ();

lcd\_put\_cur(0, 0);

lcd\_send\_string("HELLO ");

lcd\_put\_cur(1, 0);

lcd\_send\_string("UERS");

HAL\_Delay(3000);

lcd\_clear();

while (1)

{

for (int i=0; i<31; i++)

{

lcd\_put\_cur(row, col);

col++;

if (col > 15)

{

row++;

col = 0;

}

if (row > 1) row=0;

HAL\_Delay(1);

lcd\_clear();

}

if(khoitao==1) // KHI AN NUT MO CUA

{

lcd\_put\_cur(0, 0);

lcd\_send\_string(" DANG MO");

lcd\_put\_cur(1, 0);

lcd\_send\_string(" <<- ->>");

HAL\_Delay(5000);

lcd\_clear();

khoitao=0;

}

IR\_level = HAL\_GPIO\_ReadPin(GPIOA, GPIO\_PIN\_9);

if(IR\_level == 1&& tinhieu==1) // ko co vat can

{

HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_8, RESET);

HAL\_Delay(500);

demdong++;

lcd\_put\_cur(0, 0);

lcd\_send\_string(" DANG DONG");

lcd\_put\_cur(1, 0);

lcd\_send\_string(" ->> <<-");

HAL\_Delay(500);

lcd\_clear();

demmo=0;

}

if(demdong==10) // tức là đã đóng đc 5s, đong hẳn

{

lcd\_put\_cur(0, 0);

lcd\_send\_string(" DA DONG");

HAL\_Delay(3000);

lcd\_clear();

tinhieu=0;

}

if(IR\_level == 0 && tinhieu==1&&demmo==0) // CO VAT CAN

{

demdong=0;

lcd\_put\_cur(0, 0);

lcd\_send\_string(" DANG MO");

lcd\_put\_cur(1, 0);

lcd\_send\_string(" <<- ->>");

HAL\_Delay(5000);

lcd\_clear();

demmo=1;

}

if(IR\_level == 0 && tinhieu==1 && demmo > 0) // Can qua lau

{

HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_8, SET);

HAL\_Delay(500);

lcd\_put\_cur(0, 0);

lcd\_send\_string(" CO VAT CAN");

lcd\_put\_cur(1, 0);

lcd\_send\_string(" < !!!! >");

HAL\_Delay(1000);

lcd\_clear();

}

}

}

void SystemClock\_Config(void)

{

RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};

RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSE;

RCC\_OscInitStruct.HSEState = RCC\_HSE\_ON;

RCC\_OscInitStruct.HSEPredivValue = RCC\_HSE\_PREDIV\_DIV1;

RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_ON;

RCC\_OscInitStruct.PLL.PLLSource = RCC\_PLLSOURCE\_HSE;

RCC\_OscInitStruct.PLL.PLLMUL = RCC\_PLL\_MUL9;

if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)

{

Error\_Handler();

}

RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK

|RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;

RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_PLLCLK;

RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;

RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV2;

RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;

if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_2) != HAL\_OK)

{

Error\_Handler();

}

}

static void MX\_TIM1\_Init(void)

{

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

htim1.Instance = TIM1;

htim1.Init.Prescaler = 72-1;

htim1.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim1.Init.Period = 0xffff-1;

htim1.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim1.Init.RepetitionCounter = 0;

htim1.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

if (HAL\_TIM\_Base\_Init(&htim1) != HAL\_OK)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

if (HAL\_TIM\_ConfigClockSource(&htim1, &sClockSourceConfig) != HAL\_OK)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

if (HAL\_TIMEx\_MasterConfigSynchronization(&htim1, &sMasterConfig) != HAL\_OK)

{

Error\_Handler();

}

}

static void MX\_GPIO\_Init(void)

{

GPIO\_InitTypeDef GPIO\_InitStruct = {0};

/\* GPIO Ports Clock Enable \*/

\_\_HAL\_RCC\_GPIOC\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_13, GPIO\_PIN\_RESET);

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3|GPIO\_PIN\_4

|GPIO\_PIN\_5|GPIO\_PIN\_6|GPIO\_PIN\_7|GPIO\_PIN\_8, GPIO\_PIN\_RESET);

/\*Configure GPIO pin : PC13 \*/

GPIO\_InitStruct.Pin = GPIO\_PIN\_13;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(GPIOC, &GPIO\_InitStruct);

/\*Configure GPIO pins : PA1 PA2 PA3 PA4

PA5 PA6 PA7 PA8 \*/

GPIO\_InitStruct.Pin = GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3|GPIO\_PIN\_4

|GPIO\_PIN\_5|GPIO\_PIN\_6|GPIO\_PIN\_7|GPIO\_PIN\_8;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);

/\*Configure GPIO pin : PA9 \*/

GPIO\_InitStruct.Pin = GPIO\_PIN\_9;

GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;

GPIO\_InitStruct.Pull = GPIO\_PULLUP;

HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);

}

void Error\_Handler(void)

{

/\* USER CODE BEGIN Error\_Handler\_Debug \*/

/\* User can add his own implementation to report the HAL error return state \*/

/\* USER CODE END Error\_Handler\_Debug \*/

}

#ifdef USE\_FULL\_ASSERT

/\*\*

\* @brief Reports the name of the source file and the source line number

\* where the assert\_param error has occurred.

\* @param file: pointer to the source file name

\* @param line: assert\_param error line source number

\* @retval None

\*/

void assert\_failed(uint8\_t \*file, uint32\_t line)

{

/\* USER CODE BEGIN 6 \*/

/\* User can add his own implementation to report the file name and line number,

tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) \*/

/\* USER CODE END 6 \*/

}

#endif /\* USE\_FULL\_ASSERT \*/

- Khởi tạo

* Cài đặt cho LED tại PC3:

PC\_DDR\_DDR3 = 1; - Cài đặt chân PC3 là output.

PC\_CR1\_C13 = 1; - Chọn chế độ push-pull cho PC3.

* Cài đặt cho loa tại PC4:

PC\_DDR\_DDR4 = 1; - Cài đặt chân PC4 là output.

PC\_CR1\_C14 = 1; - Chọn chế độ push-pull cho PC4.

* Cài đặt cho cảm biến khí tại PD2:

PD\_DDR\_DDR2 = 0; - Cài đặt chân PD2 là input.

PD\_CR1\_C12 = 1; - Kích hoạt pull-up resistor trên PD2.

* Cài đặt Timer 4 (TIM4)

TIM4\_PSCR = 0x03; - Thiết lập prescaler cho TIM4 là 8

TIM4\_EGR\_UG = 1; - Tạo một sự kiện cập nhật để prescaler mới được áp dụng.

TIM4\_CR1\_CEN = 1; - Kích hoạt TIM4.

CPU clock = HSI (High Speed Internal)/8 = 2MHz

TIM4 mỗi 4us sẽ đếm 1 lần => mỗi 1ms sẽ đếm 250 lần

* Vòng lặp chính (Main Loop)

PC\_ODR\_ODR4 = 0; - Tắt loa.

Kiểm tra trạng thái của cảm biến:

if (PD\_IDR\_IDR2) - Nếu cảm biến khí phát hiện khí (input là HIGH):

PC\_ODR\_ODR3 = 1; - Bật LED.

PC\_ODR\_ODR4 = 1; - Bật loa.

Dealy bằng cách sử dụng TIM4 và đếm 250 lần nhịp với 100 lần lặp (100ms).

PC\_ODR\_ODR3 = 0; - Tắt LED.

Delay trong 50ms.

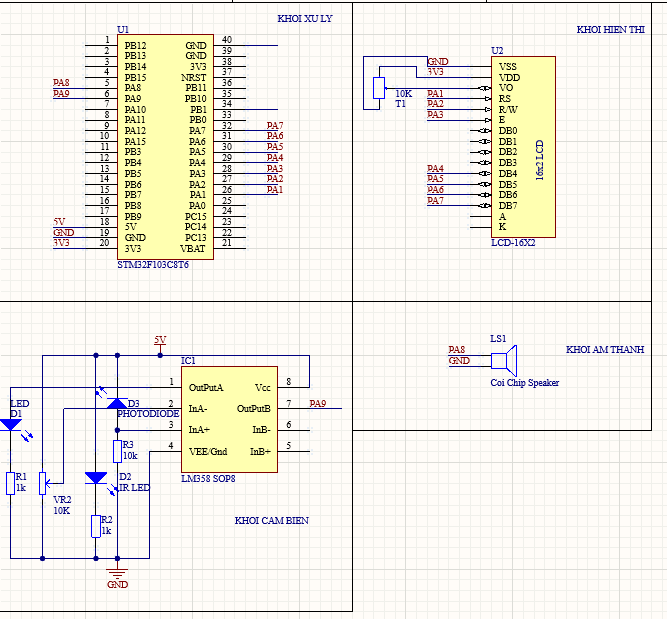
- Sau khi lập trình xong, sử dụng mạch nạp ST-Link V2 để nạp chương trình cho vi điều khiển.

## **3. Kết luận chương**

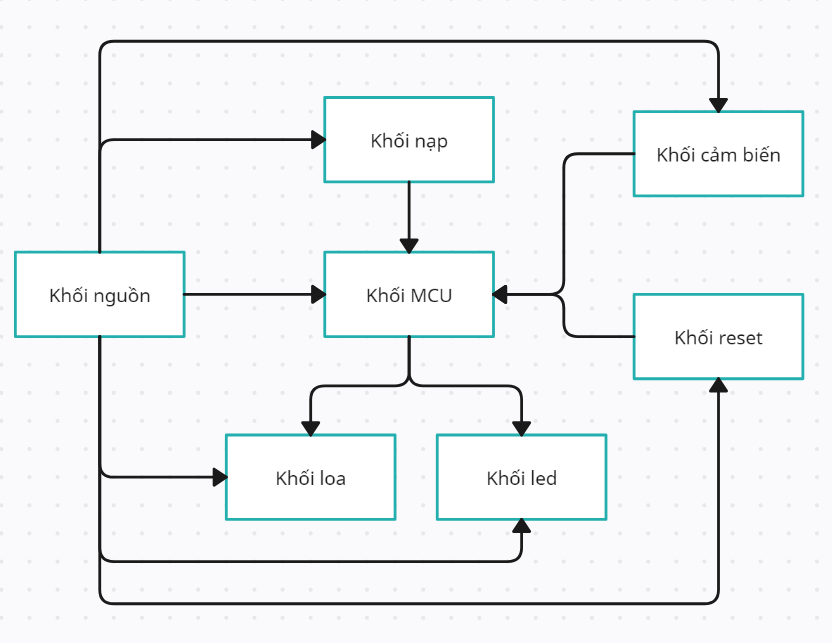
- Chương 3 này em đã nêu ra lưu đồ thuật toán, phần code và giải thích code cũng như cách nạp chuơng trình cho vi điều khiển. Đến chương cuối em xin được trình bày về sơ đồ nguyên lý và thiết kế PCB cho hệ thống.

# **CHƯƠNG 4: THIẾT KẾ PCB**

## **1. Sơ đồ nguyên lý**



*Hình 6: Sơ đồ nguyên lý*



*Hình 7: Sơ đồ khối*

- Các khối:

* Khối MCU: Vi điều khiển STM8S103F3P6
* Khối nạp: Chân pin để kết nối với ST-Link V2
* Khối cảm biến: Module MQ2
* Khối nguồn: Cổng Micro USB kết nối với AMS1117-3.3V để cấp nguồn
* Khối Led, loa: Các led hiển thị trạng thái và loa cảnh báo

(ALERT-Led cảnh báo; PWR-Led báo nguồn; TST-Led hiển thị trạng thái)

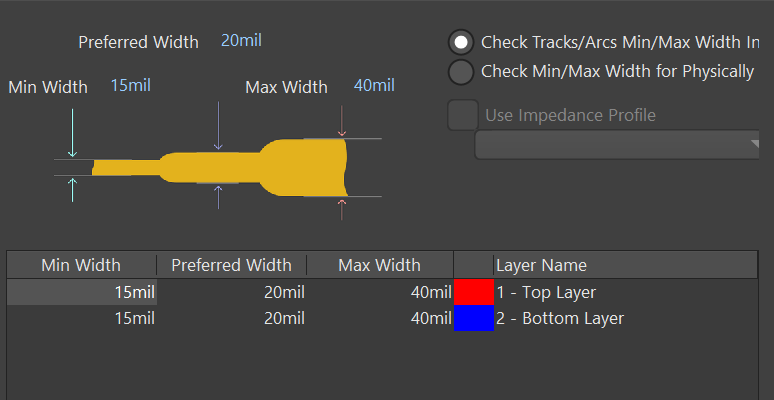
* Khối reset: nút reset của mạch

- Đèn led có thể sử dụng loại 5mm hoặc 3mm với trở tương ứng dựa vào điện áp cung cấp, kích thước và màu sắc của led.

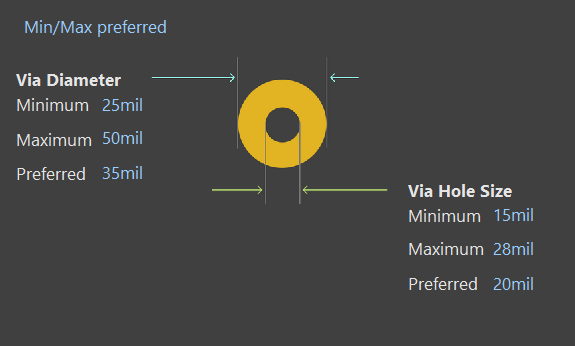
## **2. Layout của PCB**

- Sử dụng phần mềm Altium Designer để vẽ sơ đồ nguyên lý và layout cho mạch.

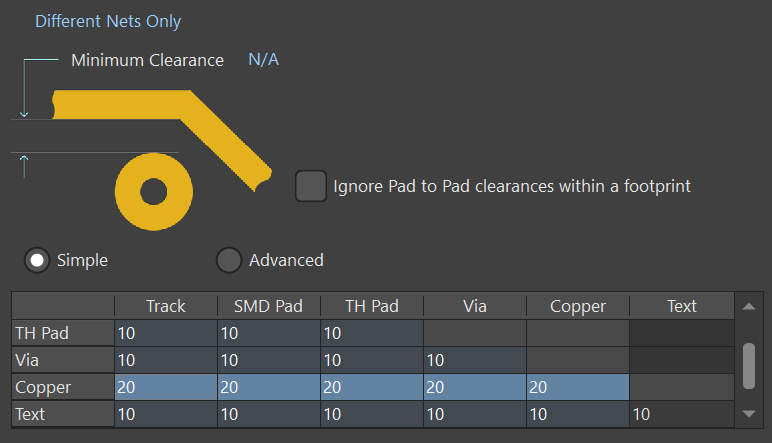
- Setting rule:



*Hình 8: Width*

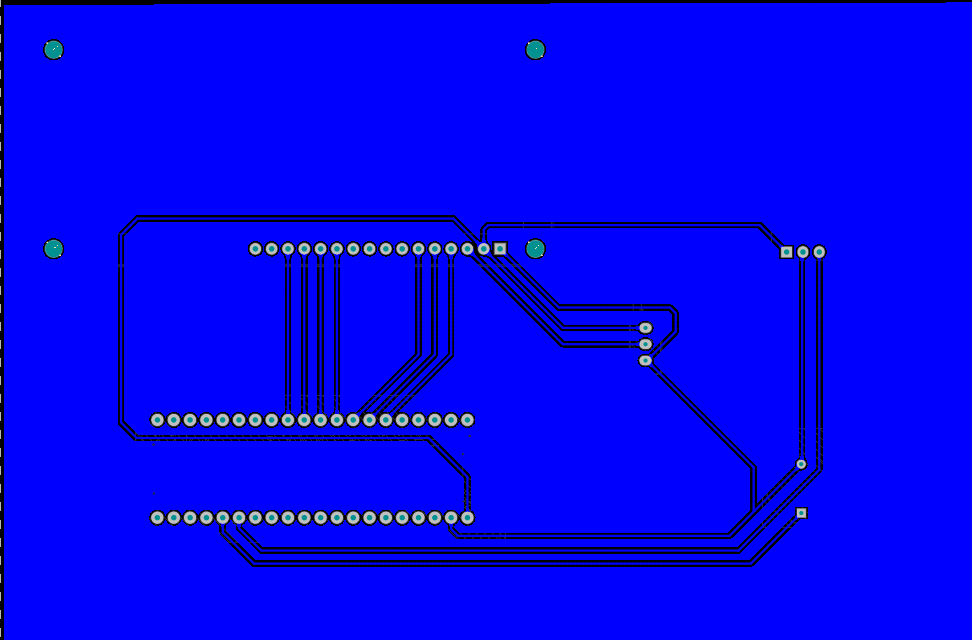


*Hình 9: RoutingVias*



*Hình 10: Clearance*

- Minimum Clearance tất cả là 10mil trừ Copper là 20mil.



*Hình 11: Đi dây trên Top Layer*