LỜI NÓI ĐẦU

Ngày nay với sự phát triển không ngừng của kĩ thuật vi điều khiển đang có những ứng dụng ngày càng rộng rãi và thâm nhập ngày càng nhiều trong các lĩnh vực kỹ thuật và đời sống xã hội nhằm ứng dụng dụng, phục vụ nhu cầu và nâng cao chất lượng cuộc sống của con người. và để ứng dụng những kiến thức đã được học thông qua các môn học trên trên lớp và sự tìm hiểu của bản thân thông qua các tài liệu sách báo và internet đó chính là mục đích của đồ án này.

Trong đồ án kỹ thuật vi điều khiển và kết nối ngoại vi này, em chọn đề là "**LED 3D cube 8x8x8**" là một ứng dụng của vi điều khiển 8051 đề tài này ứng dụng vào mục đích chủ yếu là trang trí, quảng cáo phục vụ trong cuộc sống của con người.

Trong quá trình làm đồ án, do thời gian cũng như trình độ hiểu biết của em còn hạn chế nên xảy ra nhiều sai sót là điều không thể tránh khỏi.Rất mong thầy góp ý bổ sung để em hoàn thiện hơn về kiến thức cũng như rút kinh nghiệm cho các đề tài sau.
Em xin chân thành cảm ơn !!

Đà Nẵng- 6/2015

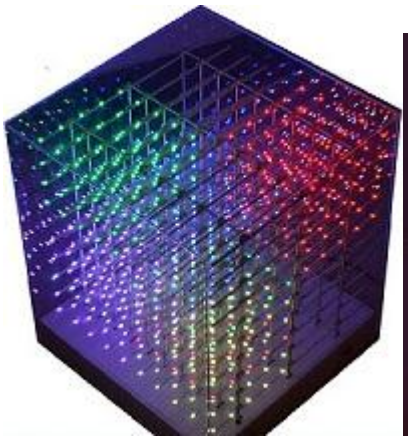
GVHD: Th.s Lê Xứng
SVTH: Nguyễn Trọng Tuấn Anh

MỤC LỤC

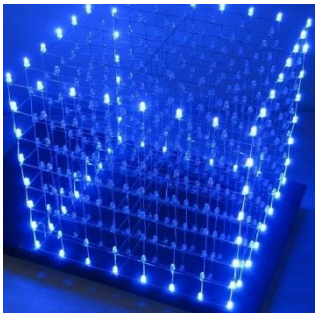
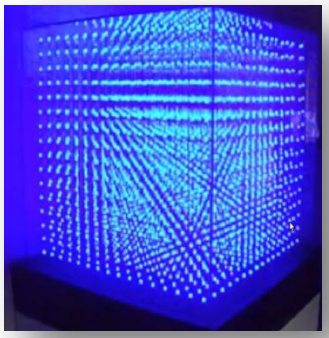
CHƯƠNG 1
GIỚI THIỆU ĐỀ TÀI

Các biển quảng cáo LED, hay các màn hình LED 2D, LED MATRIX được nhìn thấy ở khắp mọi nơi nó sữ dụng để hiện thị các hình ảnh, chữ cái ... Chúng phục vụ vào các mục đích khác nhau chẳng hạn như quảng cáo, trang trí ở các vũ trường, nhà hàng, quán café, hay là hiển thị những thông tin, hình ảnh ở các siêu thị, công ty … Tuy nhiên các hình ảnh, hiệu ứng hiện thị vẫn còn hạn chế, thiếu trực quan và ít hấp dẫn…

Để giải quyết vấn đề đó giải pháp là khối **LED 3D CUBE** có khả năng hiển thị chữ, hình ảnh theo dạng 2 chiều và 3 chiều sinh động khác biệt và ưu thế hơn so với LED 2D thông thường chúng tạo ra hình ảnh trong không gian 3 chiều hiển thị các hiệu ứng từ đơn giản đến phức tạp với nhiều kiểu phong phú, đẹp mắt … không bị gò bó và giới hạn bởi không gian phẳng mang lại tính trực quan hơn.



Khối **LED 3D CUBE** có rất nhiều loại với các kích thước và màu sắc đa dạng khác nhau phụ thuộc vào thiết kế, nhu cầu củ của chúng ta, ví dụ thông dụng nhất là loại 5x5x5, 8x8x8, 16x16x16 … Tương ứng với 125, 512, 4096 điểm ảnh…

Điều dĩ nhiên là khối LED các nhiều điểm ảnh thì việc hiện thị hình ảnh càng sắc nét, và đẹp mặt hơn có khoảng nhiều không giản rộng lớn hơn, tuy nhiên nó cũng kéo theo nhiều vấn đề phức tap như phần cứng, chương trình …

Và để thuận tiện cho việc thiết kế phần cứng cũng như lập trình, phù hợp loại vi điều khiển 8-bit thông dụng hiện này (cụ thể là 8051) thì trong đồ án này kích thước khối LED em thiết kế là 8x8x8 đơn sắc

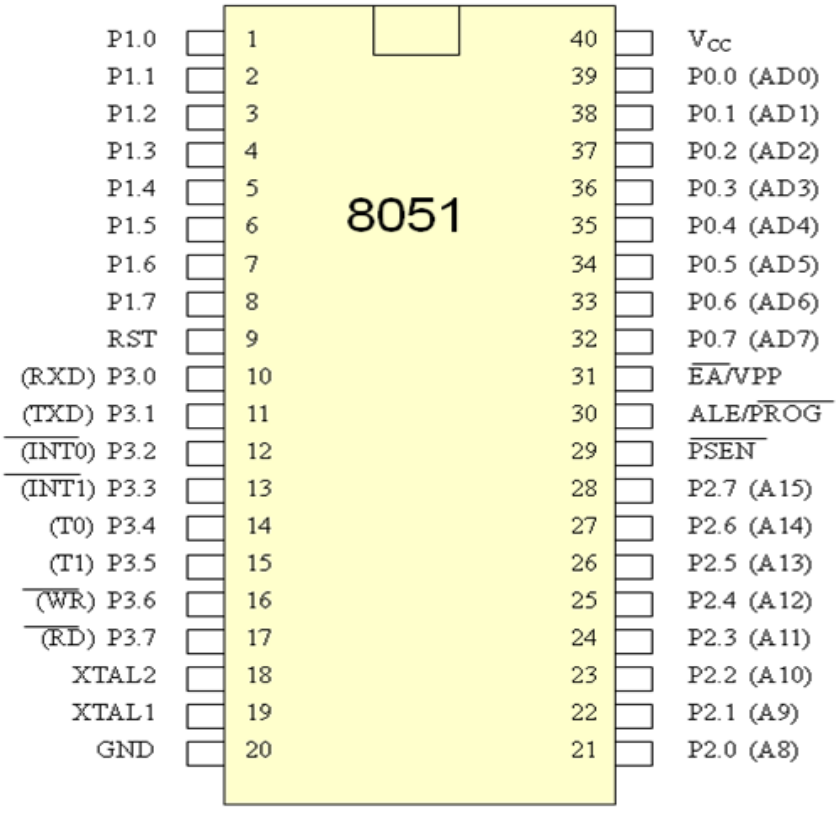
Với kích thước này cũng không quá nhỏ, hay quá lớn với 512 điểm ảnh phù hợp hiển thị những hiệu ứng đẹp mắt. Mà lại hình ảnh hài lòng cho người dùng.

CHƯƠNG 2
SƠ LƯỢC VỀ HỌ VI ĐIỀU KHIỂN 8051 VÀ VI ĐIỀU KHIỂN 89S52

2.1 Giới thiệu về họ vi điều khiển 8051

Vào năm 1980 Intel công bố chip 8051(80C51), bộ vi điều khiển đầu tiên của họ vi điều khiển MCS-51. Nó bao gồm 4KB ROM, 128 byte RAM, 32 đường xuất nhập, 1 port nối tiếp và 2 bộ định thời 16 bit. Tiếp theo sau đó là sự ra đời của chip 8052, 8053, 8055 với nhiều tính năng được cải tiến.

Hiện nay Intel không còn cung cấp các loại vi điều khiển họ MSC-51 nữa, thay vào đó là các nhà sản xuất khác như Atmel, Philips/signetics, AMD, Siemens, Matra & Dallas, Semiconductor được cấp phép làm nhà cung cấp thứ hai cho các chip họ MSC-51. Chúng được ứng dụng rộng rãi trên thế giới cũng như ở Việt Nam hiện nay là Vi điều khiển của hãng Atmel.



Chúng có tính năng như sau :

- 4k byte ROM (được lập trình bởi nhà sản xuất, chỉ có trong 8051).

- 128 byte RAM.

- 4 Port I/O 8 bit.

- 2 bộ định thời 16 bit.

- Giao tiếp nối tiếp.

- 64K không gian bộ nhớ chương trình mở rộng.

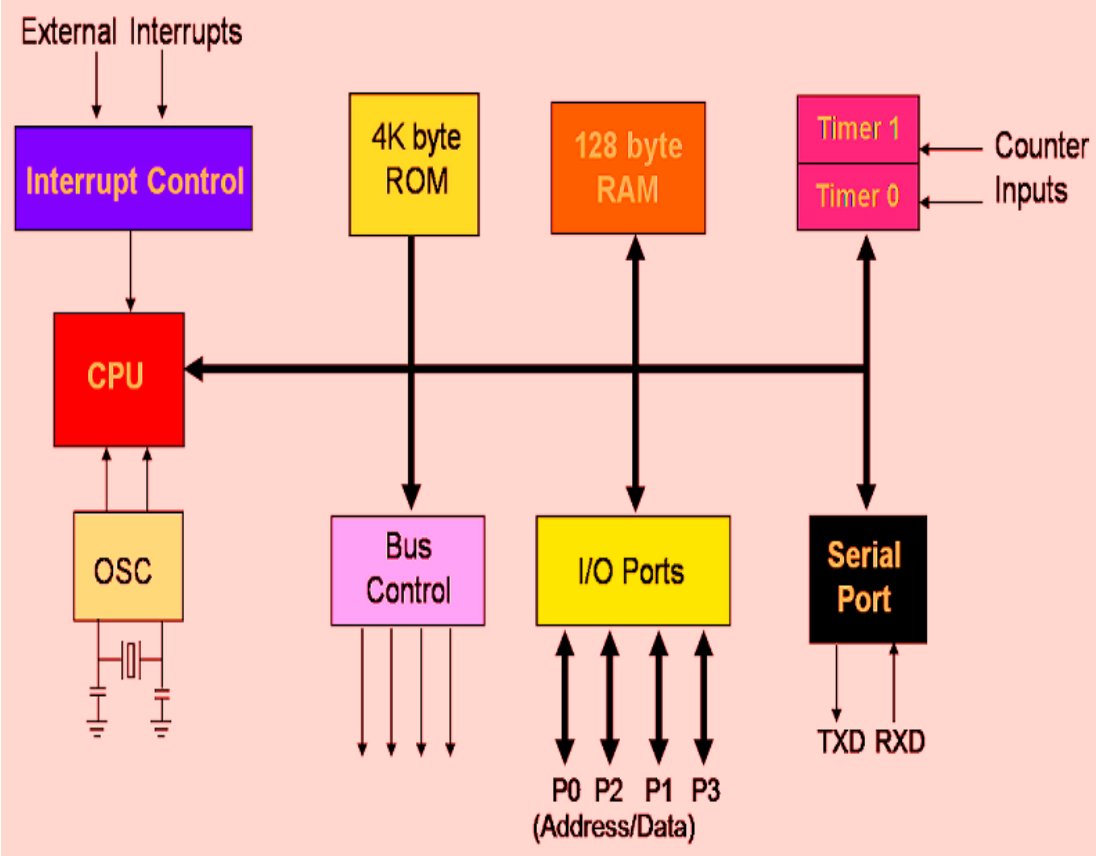
- 64K không gian bộ nhớ dữ liệu mở rộng.

- Một bộ xử lý luận lý (thao tác trên các bit đơn).

- 210 bit được địa chỉ hóa.

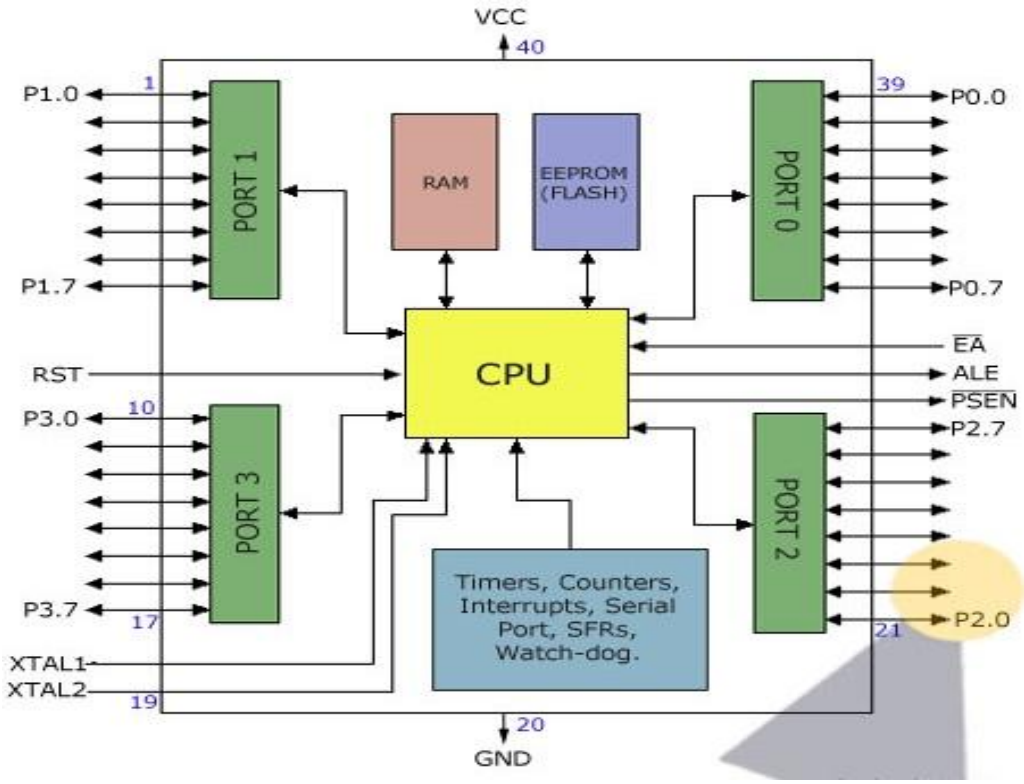
- Bộ nhân /chia 4 bit.

*** Cấu trúc bên trong :**



***** CPU(*CPU centralprocessing unit)* bao gồm:

- Thanh ghi tích lũy A
- Thanh ghi tích lũy phụ B
- Đơn vị logic học (ALU)
- Thanh ghi từ trạng thái chương trình
- Bốn băng thanh ghi
- Con trỏ ngăn xếp
- Bộ nhớ chương trình(ROM) gồm 8Kbyte Flash.
- Bộ nhớ dữ liệu(RAM) gồm 256 byte.
- Bộ UART, có chức năng truyền nhận nối tiếp.
- 3 bộ Timer/Counter 16 bit thực hiện chức năng định thời và đếm sự kiện.
- Khối điều khiển ngắt với 2 nguồn ngắt ngoài và 4 nguồn ngắt trong.
- Bộ lập trình(ghi chương trình lên Flash ROM) cho phép người sử dụng có thể nạp các chương trình cho chíp mà không cần các bộ nạp chuyên dụng.
- Bộ chia tần số với hệ số chia là 12.
- 4 cổng xuất nhập với 32 chân.



❖ Port 0(P0.0=>P0.7): Port 0 gồm 8 chân, ngoài chức năng xuất nhập, port 0 còn là bus đa hợp dữ liệu và địa chỉ(AD0-AD7), chức năng này sẽ được sử dụng khi 89c52 giao tiếp với các thiết bị ngoài có kiến trúc Bus như các vi mạch nhớ, mạch PIO…

❖ Port 1(P1.0=>P1.7): Chức năng duy nhất củaPort 1 là chức năng xuất nhập cũng như cácPort khác. Port1 có thể xuất nhập theo bit và theo byte.

❖ Port 2(P2.0=>P2.7); Port 2 ngoài chức năng là cổng vào/ra nhưPort 0 và 1 còn là byte cao của bus địa chỉ khi sử dụng bộ nhớ ngoài.

❖ Port 3: Mỗi chân trên Port 3 ngoài chức năng xuất nhập còn có một chức năng riêng, cụ thể
như sau:

Bit	Tên	Chức năng
P3.0	RXD	Dữ liệu nhận choPort nối tiếp
P3.1	TXD	Dữ liệu truyền choPort nối tiếp
P3.2	INT0	Ngắt bên ngoài 0
P3.3	INT1	Ngắt ngoài 1
P3.4	TO	Ngõ vào của Timer/counter0

P3.5	T1	Ngõ vào của Timer/counter1
P3.6	/WR	Xung ghi bộ nhớ dữ liệu ngoài.
P3.7	/RD	Xung đọc bộ nhớ dữ liệu ngoài.

❖ Chân /PSEN : là chân điều khiển đọc chương trình ở bộ nhớ ngoài.

❖ Chân ALE: ALE là tín hiệu điều khiển chốt địa chỉ có tần số bằng 1/6 tần số dao động của vi điều khiển. Tín hiệu ALE được dùng để cho phép vi mạch chốt bên ngoài như 7473.

❖ Chân /EA: Tín hiệu /EA cho phép chọn bộ nhớ chương trình là bộ nhớ trong hay ngoài. EA=1 thì thực hiện chương trình trong RAM nội. EA=0 thực hiện ở RAM ngoài.

❖ RST(reset): Ngõ vào reset trên chân số 9. khi RST=1 thì bộ vi điều khiển sẽ được khởi động lại thiết lập ban đầu.

❖ XTAL1, XTAL2: 2 chân này được nối song song với thạch anh tần số max=33 Mhz. Để tạo dao động cho bộ vi điều khiển.

❖ Vcc, GND : cung cấp nguồn nuôi cho bộ vi điều khiển. cấp qua chân 20 và 40.

2.2 Vi điều khiển 89S52

Vi điều khiển AT89S52 thuộc họ vi điều khiển công suất thấp 8 bit của hãng Atmel với tần số hoạt động 33 MHz, với 8 Kb bộ nhớ Flash, 256 Bytes RAM, thành phần ngoại vi chỉ có Timer, Counter, và giao diện kết nối bên ngoài USART song công.

***** Họ vi điều khiển 8 bit

***** Tần số hoạt động : 33 Mhz

***** Bộ nhớ :8 Kb Flash, 256 Bytes SRAM

***** Timer/Counter : 3 bộ 16 bit

***** Giao diện kết nối : USART

***** Lập trình qua giao diện ISP

Tất cả các vi điều khiển họ 8051 đều có đặc tính cơ bản giống nhau về phần mềm (các tập lệnh lập trình như nhau), tuy nhiên vi điều khiển 89s52 vi điều khiển sau này có nhiều tính năng vượt trội hơn. Vi điều khiển 89s52 có bộ nhớ ROM, RAM lớn hơn so với vi điêu hiển 89CXX được bổ sung một số tính năng và có thêm chế độ nạp nối tiếp.

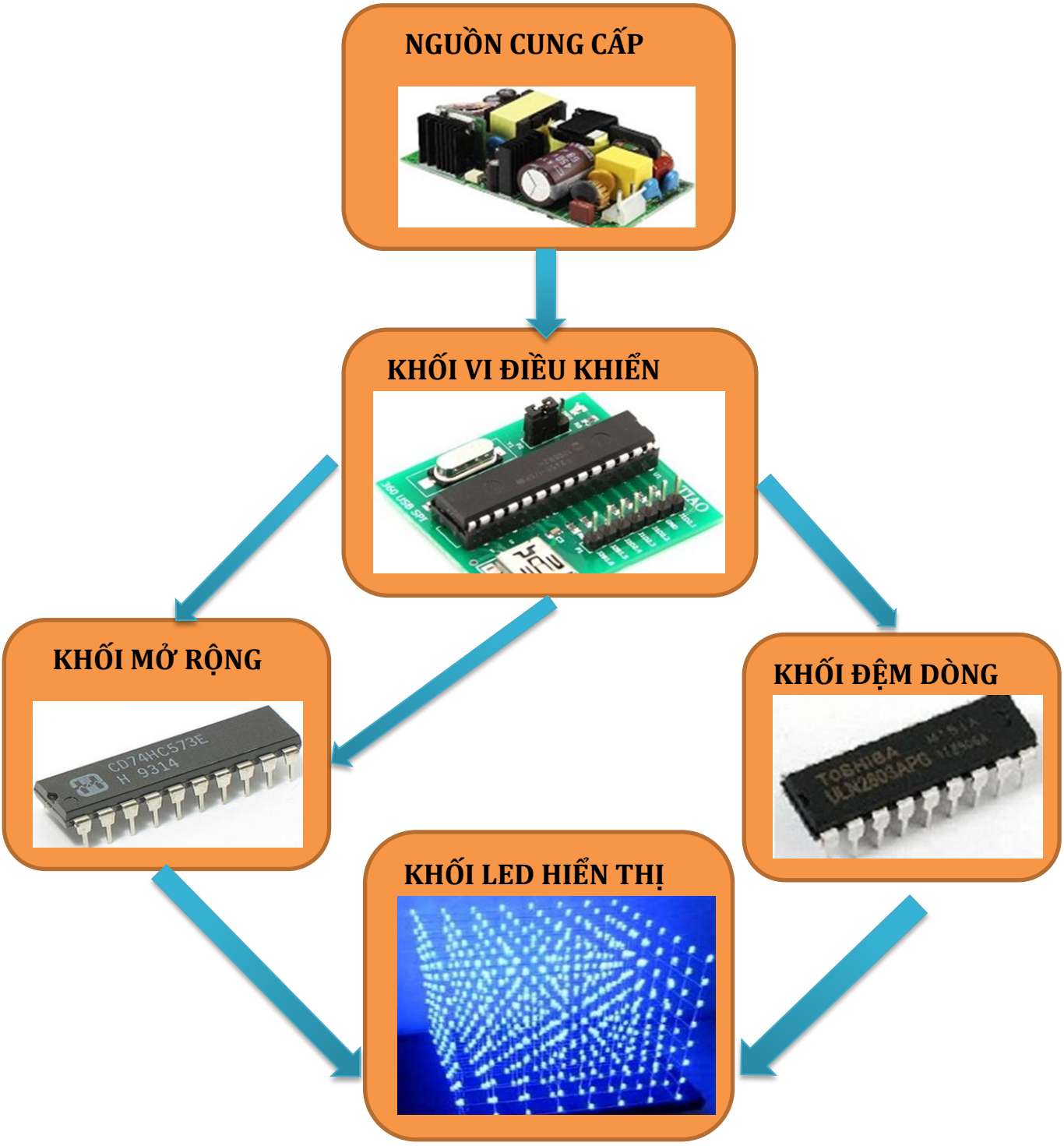
	Dung lượng RAM	Dung lượng ROM	Chế độ nạp
89S51	128 byte	4 Kbyte	nối tiếp
89S52	128 byte	8 Kbyte	nối tiếp
89S53	128 byte	12 Kbyte	nối tiếp
89S55	128 byte	20 Kbyte	nối tiếp

89S52 có chế độ nạp nối tiếp với mạch nạp đơn giản có khả năng nạp ngay trên bo mạch mà không cần tháo chip vi điều khiển sang mạch khác để nạp chương trình và nhiều tính năng cải tiến khác. Đó là lý do lựa chọn vi điều khiển 89s52.

CHƯƠNG 3

THIẾT KẾ PHẦN CỨNG VÀ THI CÔNG

3.1. Sơ đồ khối và chức năng từng khối



Chức năng từng khối:

- Nguồn cung cấp

Gồm các linh kiện như biến áp, tụ lọc, và các IC ổn áp dùng để hạ áp- chỉnh lưu- ổn định điện áp, tạo nguồn điện áp phù hợp, và ổn định nhằm cung cấp nguồn cho mạch vi điều khiển và khối LED hoạt động tốt.

- Khối vi điều khiển

Là bộ phận điều khiển trung tâm quan trọng trọng nhất trong mạch nó bao gồm phần RESET (Khi chương trình bị lỗi, nguồn cung cấp không ổn định, hay là do tác động cứng bên ngoài thì mạch sẽ được RESET lại trang thái ban đầu), dao động thạch anh (vi điều khiển 8051 sử dụng thạch anh ngoại nhằm tạo xung nhịp, tần số ổn định), các chân I/O. Mọi quá trình tính toán, tạo mã, xuất dữ liệu, quét…Sẽ được xử lý tại đây.

- Khối mở rộng

Dùng để mở rộng chân vi điều khiển dựa trên nguyên tắc dịch, chốt dữ liệu. Lý do vi điều khiển 8051 sử dụng có số lượng chân I/O tối đa là 32 chân nhưng để điều khiển được khối LED số chân cần thiết là 64 chân (xuất DATA ra mảng) + 8 chân điều khiển cung cấp nguồn cho lớp, tổng cộng có 72 chân. Trong khối LED này sử dụng 8 IC mở rộng tại một thời điểm chỉ đưa dữ liệu ra một IC, còn lại những IC khác bị chốt lại nên dữ liệu không bị ảnh hưởng, khi đã xuất DATA ra đủ 8 con ta chốt và xuất dữ liệu ra.

- Khối đệm dòng

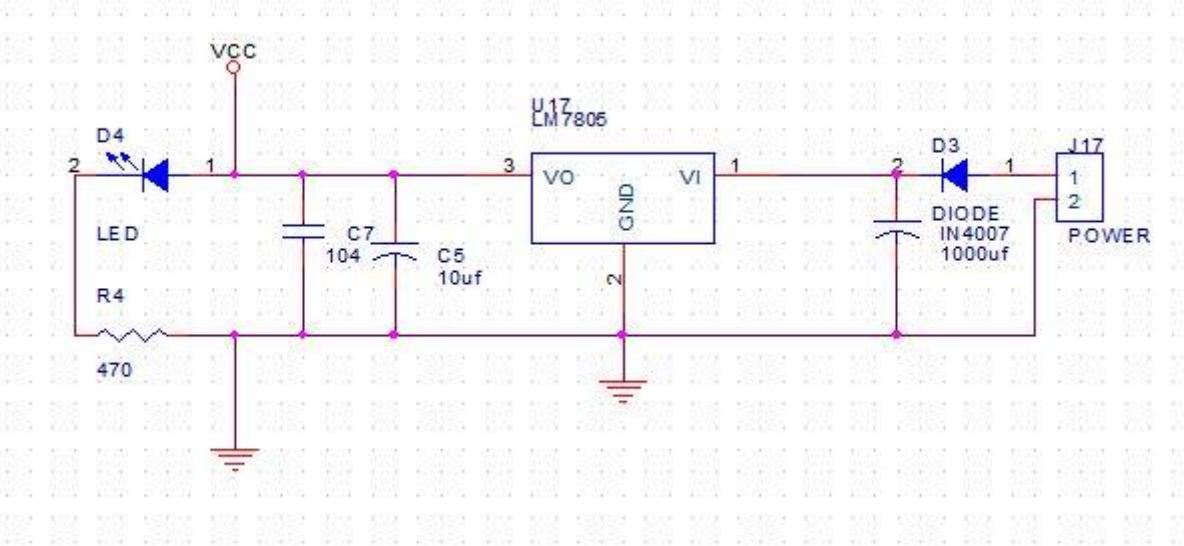
Gồm các IC đệm được tích hợp các bộ darlington bằng BJT ở bên trong vừa nhỏ gọn lại giúp thiết kệ mạch dễ dàng. Nhằm mục đích khếch đại dòng cung cấp đủ dòng để khối LED sáng hiển thị tốt.

- Khối LED hiển thị

Là phần hiển thị mà chúng ta sẽ nhìn thấy bên ngoài, mọi hiệu ứng, hình ảnh, chữ… Chúng ta sẽ được nhìn thấy ở đây.

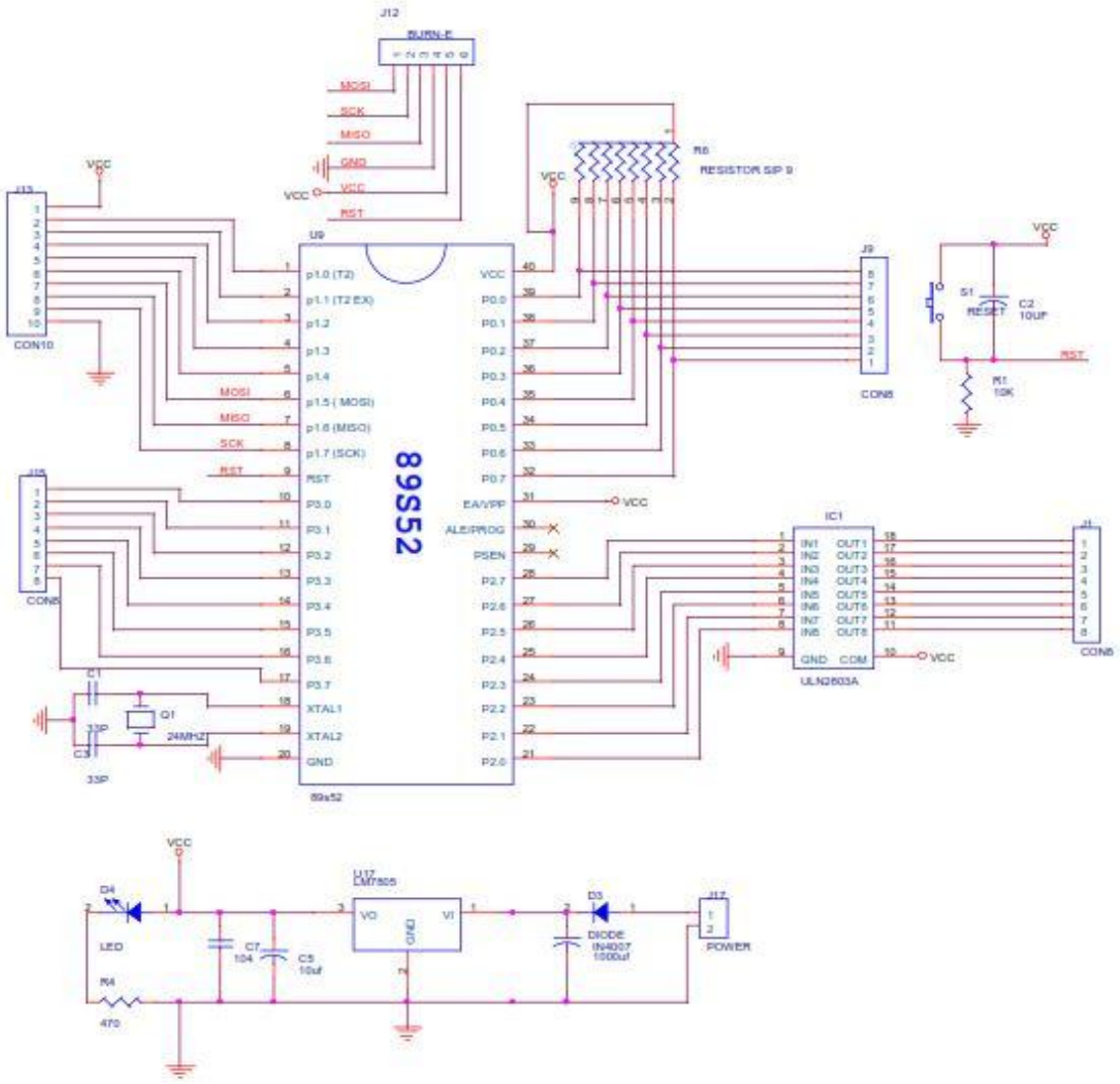
3.2 Cấu trúc phần cứng từng khối

3.2.1 Khối nguồn



- Mạch nguồn dùng để tạo ra nguồn điện áp chuẩn +5V Sử dụng IC ổn áp 7805 dùng làm nguồn nuôi vi điều khiển và khối LED hiển thị nhằm cung cấp cho điện áp phù hợp để mạch hoạt động tốt khối LED hiển thị đẹp, bền.
- Đầu vào sử dụng điện áp 9VDC sử dụng adapter , qua diode IN4007 tránh hiện tượng cắm nhầm cực gây hỏng mạch, qua tụ lọc 1000uF, sau đó đi qua IC ổn áp 7805 để tạo điện áp 5VDC

3.2.2 Khối vi điều khiển trung tâm



Là bộ phận điều khiển trung tâm quan trọng trọng nhất trong mạch nó bao gồm phần RESET (Khi chương trình bị lỗi, nguồn cung cấp không ổn định, hay là do tác động cứng bên ngoài thì mạch sẽ được RESET lại trang thái ban đầu), dao động thạch anh (vi điều khiển 8051 sử dụng thạch anh ngoại nhằm tạo xung nhịp, tần số ổn định), các chân I/O. Mọi quá trình tính toán, tạo mã, xuất dữ liệu, quét…Sẽ được xử lý tại đây.

Trong đồ án này sử dụng vi điều khiển 89s52 có bộ nhớ ROM, RAM lớn hơn so với vi điêu hiển 89CXX được bổ sung một số tính năng và có thêm chế độ nạp nối tiếp.

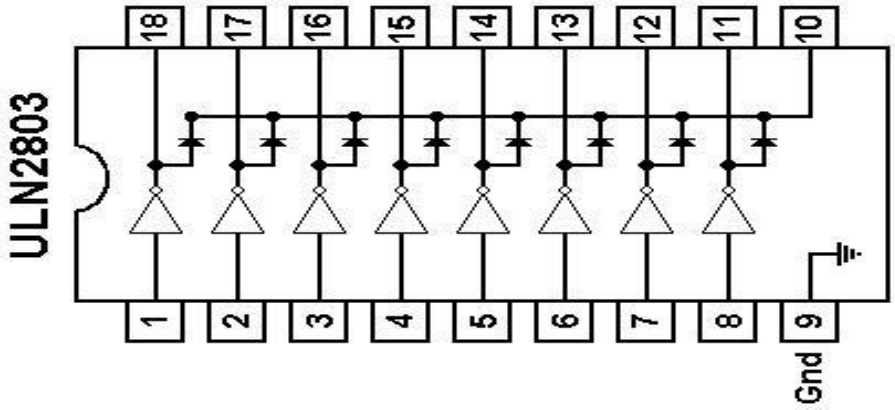
89S52 có chế độ nạp nối tiếp với mạch nạp đơn giản có khả năng nạp ngay trên bo mạch mà không cần tháo chip vi điều khiển sang mạch khác để nạp chương trình tiện lợi hơn.

-Bộ dao động thạch anh có tác dụng tạo xung nhịp với tần số 12MHz cho vi điều khiển hoạt động.

- Bộ RESET có tác RESET vi điều khiển lại trang thái ban đầu. Khi nút Reset được tác động cứng từ nút nhần và mạch reset khi ở trạng thái mức cao.

3.2.3 Khối đệm dòng

Ở đây sử dụng IC đệm đảo ULN2803 bên trọng được tích hợp 8 bộ darlington bằng BJT ở bên trong vừa nhỏ gọn lại giúp thiết kệ mạch dễ dàng. Nhằm mục đích khếch đại dòng cung cấp đủ dòng để khối LED sáng hiển thị tốt.



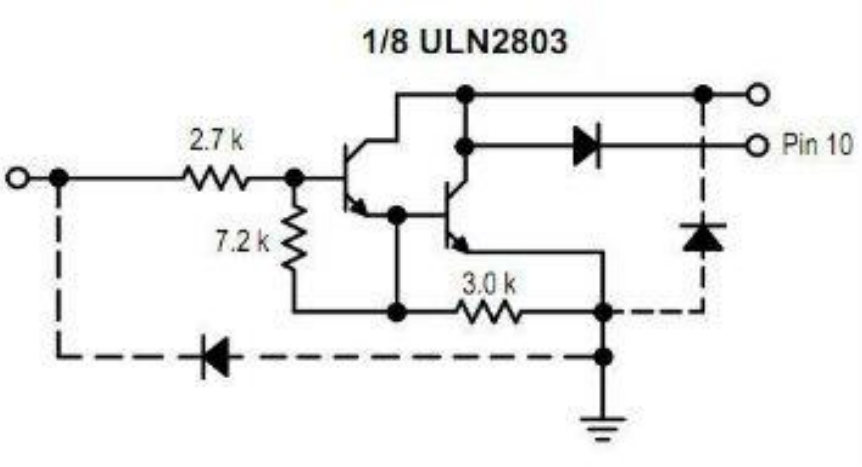
ULN2803 darlington transistors gồm 8 cặp transistor mắc theo kiểu darlington cho phép tải được dòng lên đến 500mA. ULN2803 có khả năng chịu được điện áp cao, lên đến 30V.

- Dòng điện ngỏ vào khoảng 25mA
- Điệp áp ngỏ vào khoảng 0.5-30V

- Dòng ra 500mA

- Đệm 8 kênh riêng biệt

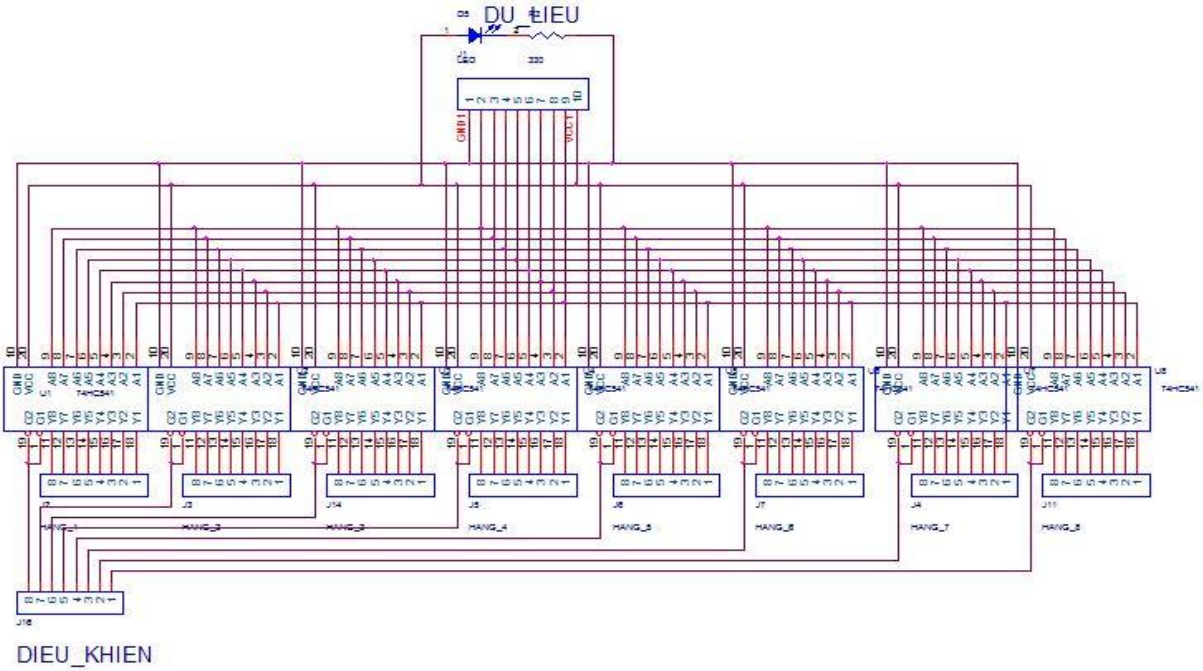
- Có đầu ra đảo



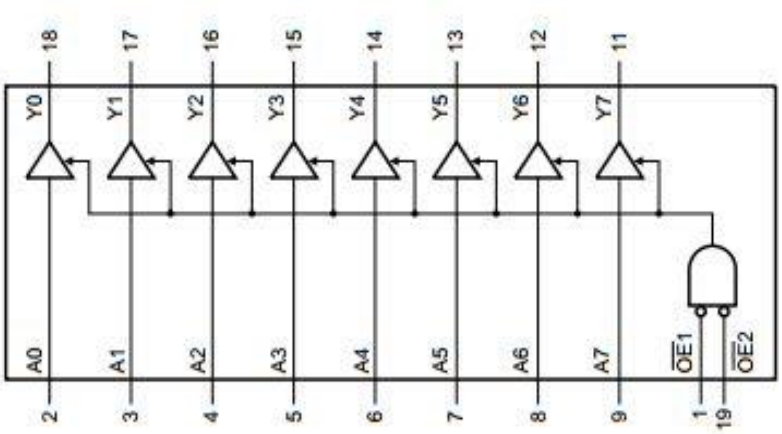
Sơ đồ cấu tạo 1 kênh đệm dòng của ULN2803

3.2.4 Khối mở rộng

- Dùng để mở rộng chân vi điều khiển dựa trên nguyên tắc chốt dữ liệu. Lý do vi điều khiển 8051 sử dụng có số lượng chân I/O tối đa là 32 chân nhưng để điều khiển được khối LED số chân cần thiết là 64 chân (xuất DATA ra mảng) + 8 chân điều khiển cung cấp nguồn cho lớp, tổng cộng có 72 chân. Trong khối LED này sử dụng 8 IC mở rộng tại một thời điểm chỉ đưa dữ liệu ra một IC, còn lại những IC khác bị chốt lại nên dữ liệu không bị ảnh hưởng, khi đã xuất DATA ra đủ 8 con ta chốt và xuất dữ liệu ra. ở đây sử dụng IC mở rộng 74HC541



-74HC541 có tất cả 8 đầu vào dử liệu tương ứng 8 đầu ra kết hợp với 2 chân chốt dư liệu là OE1 và OE2 khi cả 2 chân này ở mức thấp cho phép xuất dữ liệu ra, mọi trường hợp khác cấm xuất dữ liệu đầu ra lúc này sẽ cho trở kháng cao.



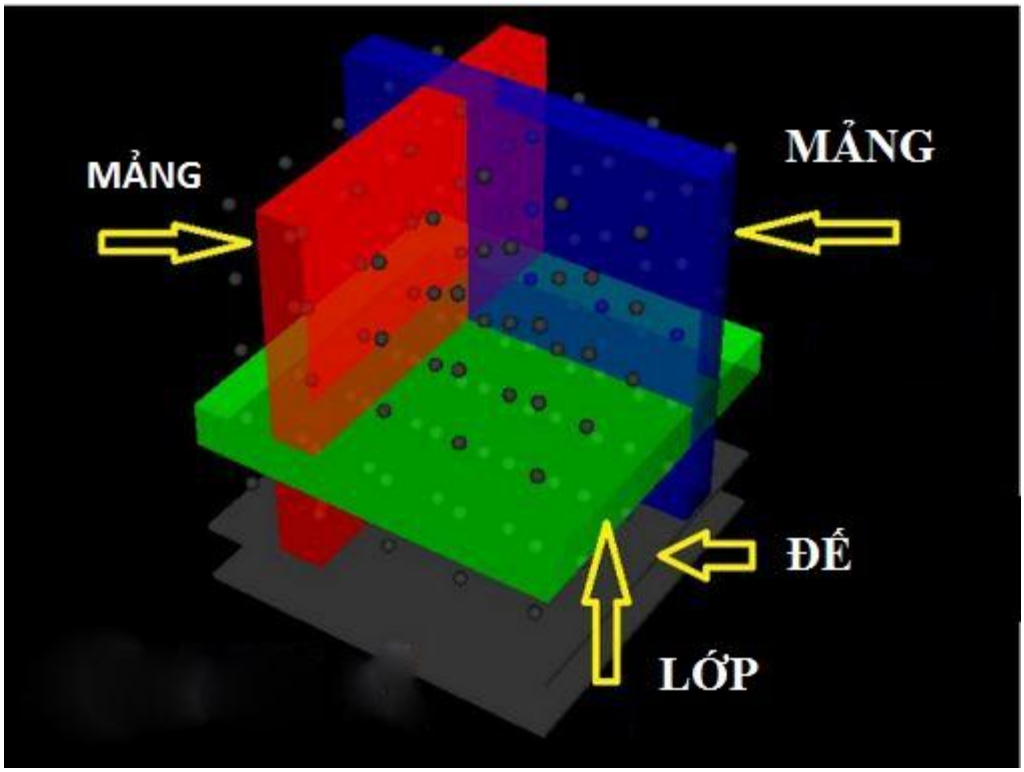
- Dòng ra của 74hc541 là 35mA điện áp hoạt động từ 2-6V

3.2.5 Khối LED hiển thị

Là phần hiển thị mà chúng ta sẽ nhìn thấy bên ngoài, mọi hiệu ứng, hình ảnh, chữ… Chúng ta sẽ được nhìn thấy ở đây.

- Khối LED hiển thị khi đã hoàn thành :

Trước tiên ta quy ước cac lớp, mảng, cột như hình vẽ để tiện quan sát.



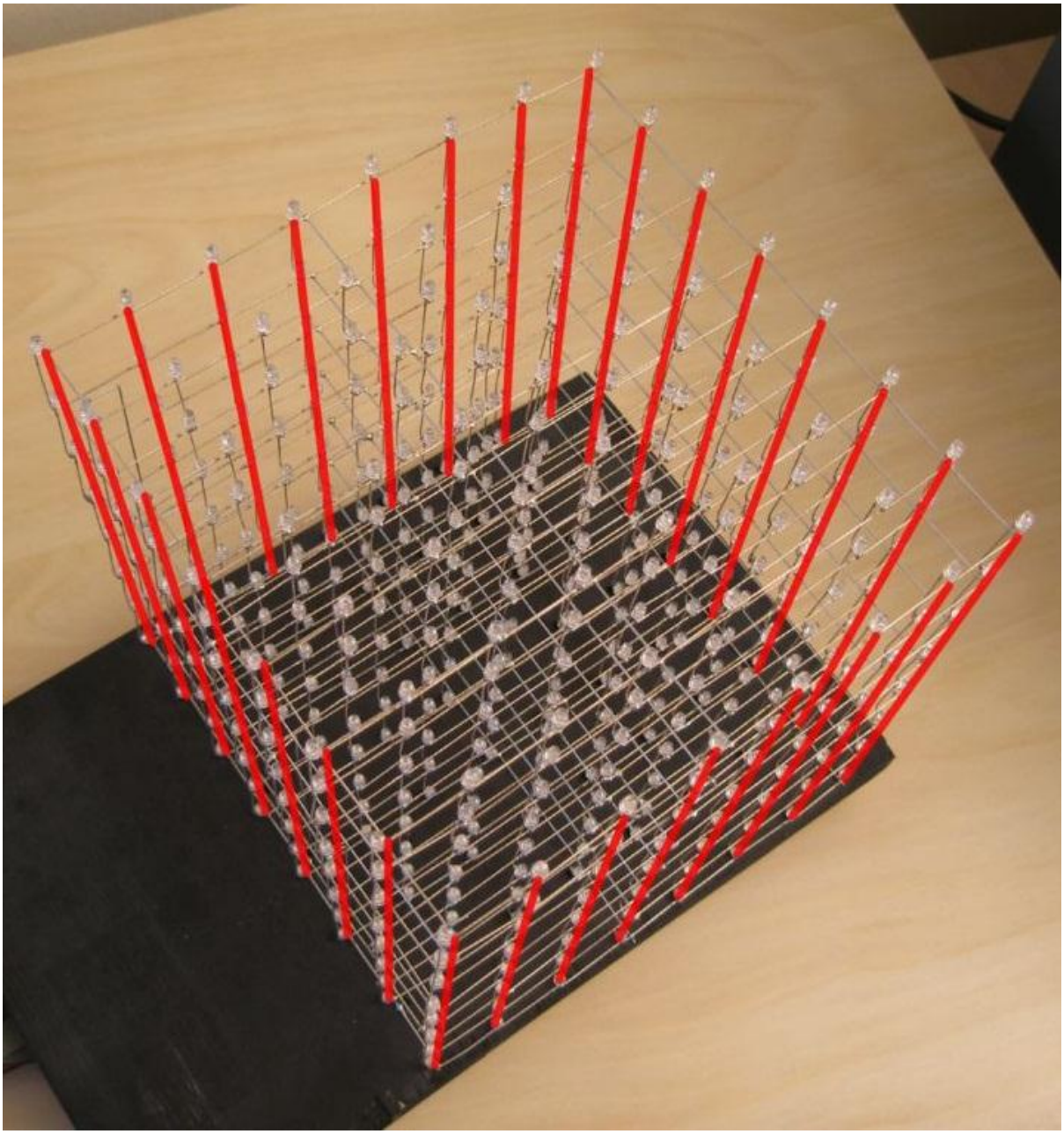
Lớp : Quy ước theo chiều từ dưới lên theo thứ tự lớp 1-8
Mảng : Theo chiều từ trước ra sau Và các cột

Kếu cấu của khối LED gồm 8 lớp 8 mảng và có 64 cột tổng cộng gồm 512 con LED được liên kết với nhau bởi những thanh thép, để liên kết khối LED vững chắc

mỗi mảng led lại ghép nhiều led với nhau. Sẽ tạo thành khối cube lập phương, hình ảnh bên dưới là ghép 1 tầng led với nhau, chân Anode (chân dài) sẽ tạo thành các cột led. các chân ngắn sẽ bẻ ngang bẻ chung lại thành các mảng chung âm

Mỗi lớp gồm 8 hàng, 8 cột LED sắp theo kiểu ma trận với 64 LED đê tạo thành lớp với Catot chung

Sắp xếp đủ 8 lớp sẽ tao thành khối LED 8 lớp với cưc - (Catot) chung sẽ được điều khiển cấp nguồn thông qua IC đệm ULN2803.

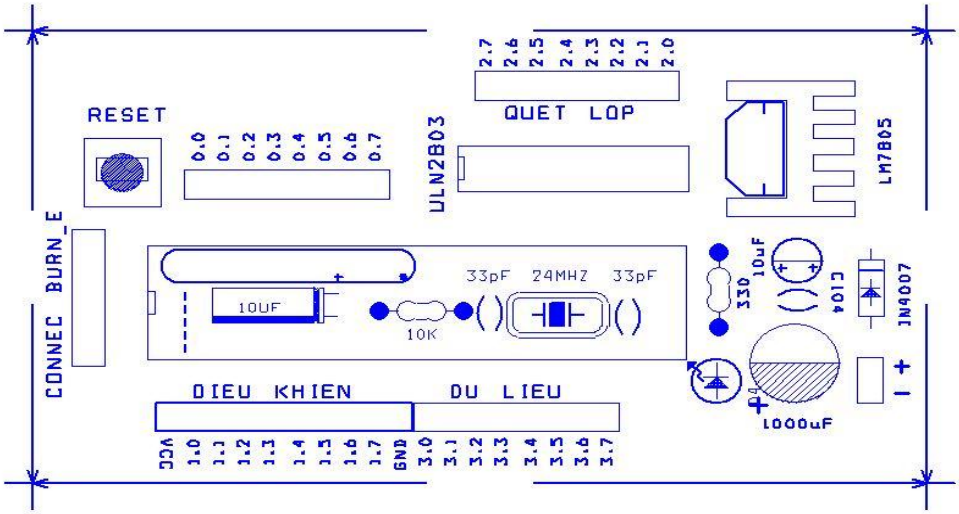


Tiến hành hàn nối các mãng LED lại với nhau sau cho các con LED cách đều nhau và hàn cố định khối LED3D lên mạch điều khiển. Khi này khối LED3D và mạch sẽ tạo thành 1 khối lập phương, bao gồm 64 Cột và 8 Lớp. Các mãng phải cách đều nhau, song song với nhau và song song với board mạch điều khiển. Các chân Âm của các con LED trên cùng 1 mãng phải được nối lại với nhau. Các điểm LED càng cách đều nhau thì càng thể hiện được hiệu ứng một cách sinh động và đẹp mắt, các hiệu ứng sẽ rõ ràng hơn.

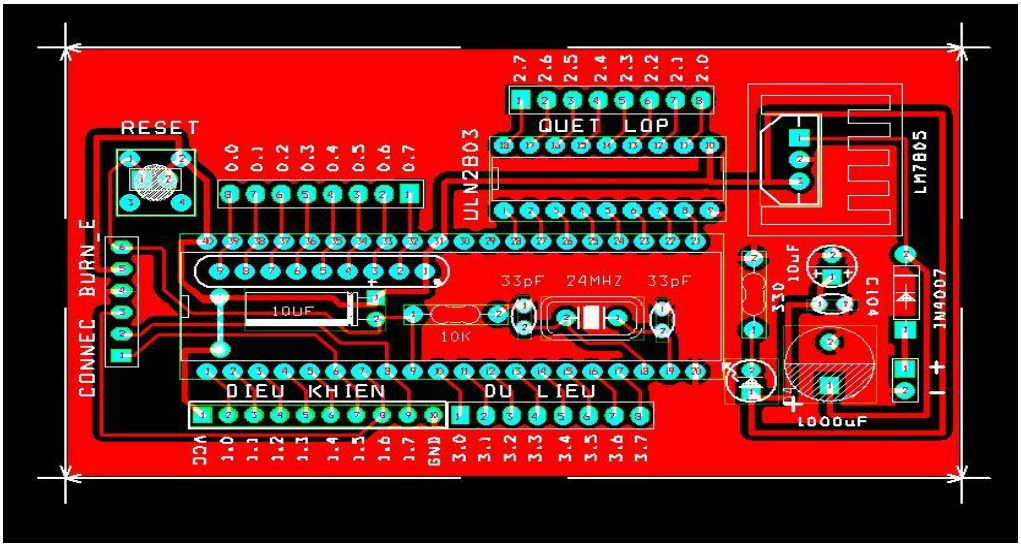
3.3 Hình ảnh mạch layout

Mạch vi điều khiển:

+ Mặt trên :

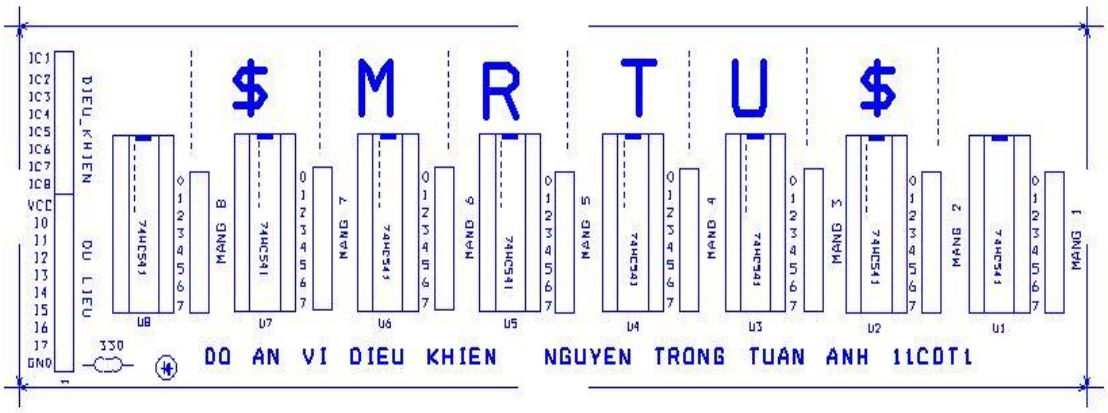


+ Mặt dưới :

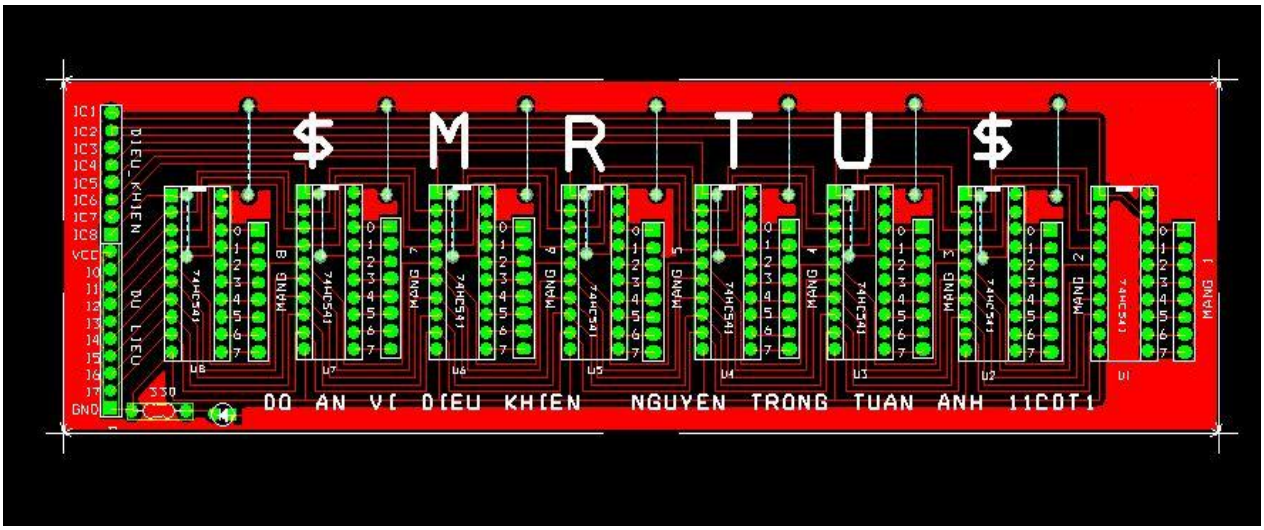


Mạch mở rộng 74hC54 :

+ Mặt trên :



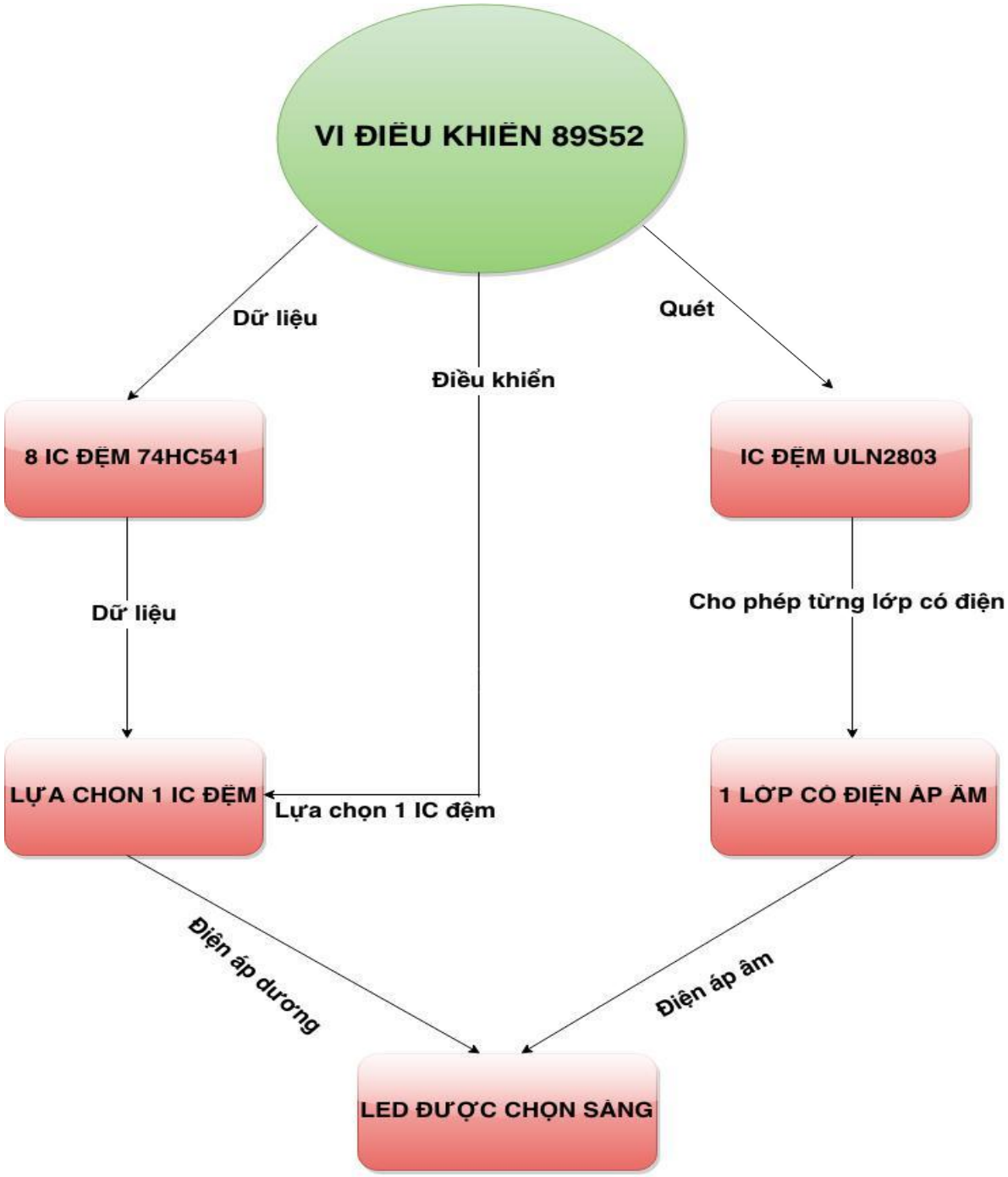
+ Mặt dưới :



CHƯƠNG 4

LƯU ĐỒ THUẬT TOÁN VÀ CHƯƠNG TRÌNH

4.1. Lưu đồ thuật toán chung



- Bộ điều khiển trung tậm là chip 89S52
Port1 của vi điều khiển cấp dữ liệu
Port2 của vi điều khiển để quét dữ liệu thông qua IC đệm ULN2803

Port3 của vi điều khiển dùng để điều khiển IC74hc541 nào được chọn

Tại 1 thời điểm chỉ có 1 IC74HC541 được chon và xuất dữ liệu ra cứ 1 IC được chon tương ứng với 1 lớp được điều khiển bởi ULN2803 được chọn khi đã xuất dư liệu và quét tương ứng với 8 lần với thời gian đủ nhanh để tạo hiện tượng lưu ảnh của mắt thì khối LED sẽ được hiện thị.

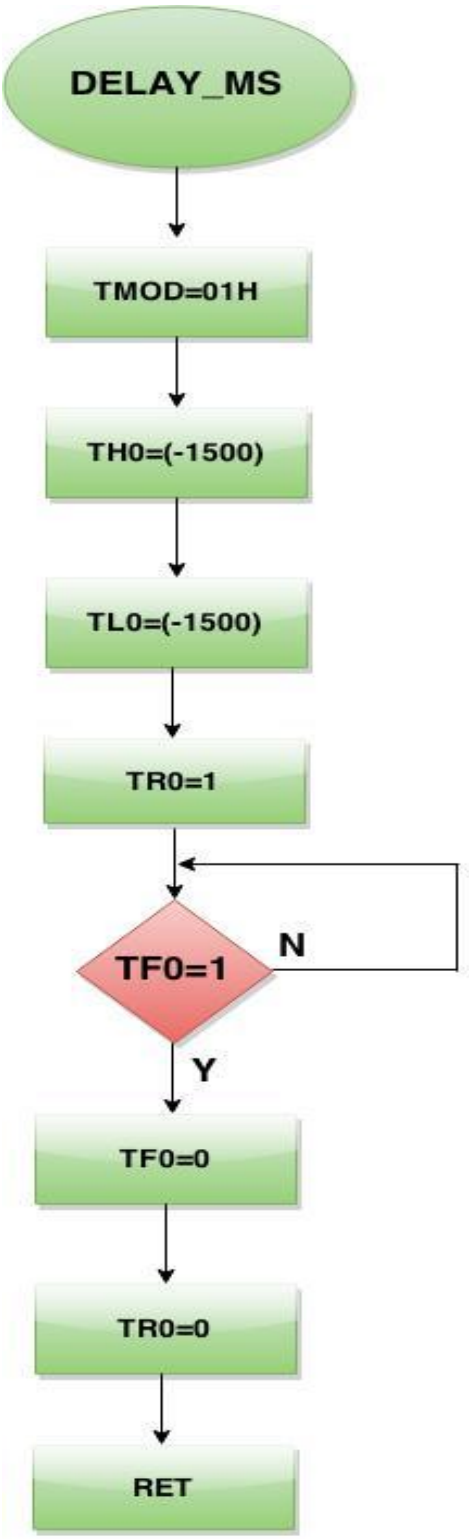
4.2. Thuật toán và một số chương trình con

4.2.1 Hàm delay_ms

- Để khối led hoat động ổn định , sáng đều và đẹp thì ngoài mạch nguồn cung câp, điều quan trọng trong chương trình đó là tần số quét. Nếu tần số quét quét led quá lơn tốc độ sáng qua nhanh led sáng mờ. Còn nếu tần số quét led quá nhỏ thì không tạo ra được hiện tượng lưu ảnh khi đó LED sẽ sáng chập chờn vì vậy ta chọn tấn số quét hợp lý, dựa vào lý thuyết và kinh nghiệm thực tế chọn tần số f=80hz .ta quét 8 lần theo từng lớp, vì vậy f=80x8.

Vậy T= 1/(80*8) =1562.5us trừ khoảng thời gian xuất dữ liệu, dịch chốt ta chọn T=1500us Vì vậy ta phải tạo 1 chương trình con delay với T=1500

Thuật toán :


DELAY_MS
TMOD=01H
TH0=(-1500)
TL0=(-1500)
TR0=1
TF0=1
N
Y
TF0=0
TR0=0
RET

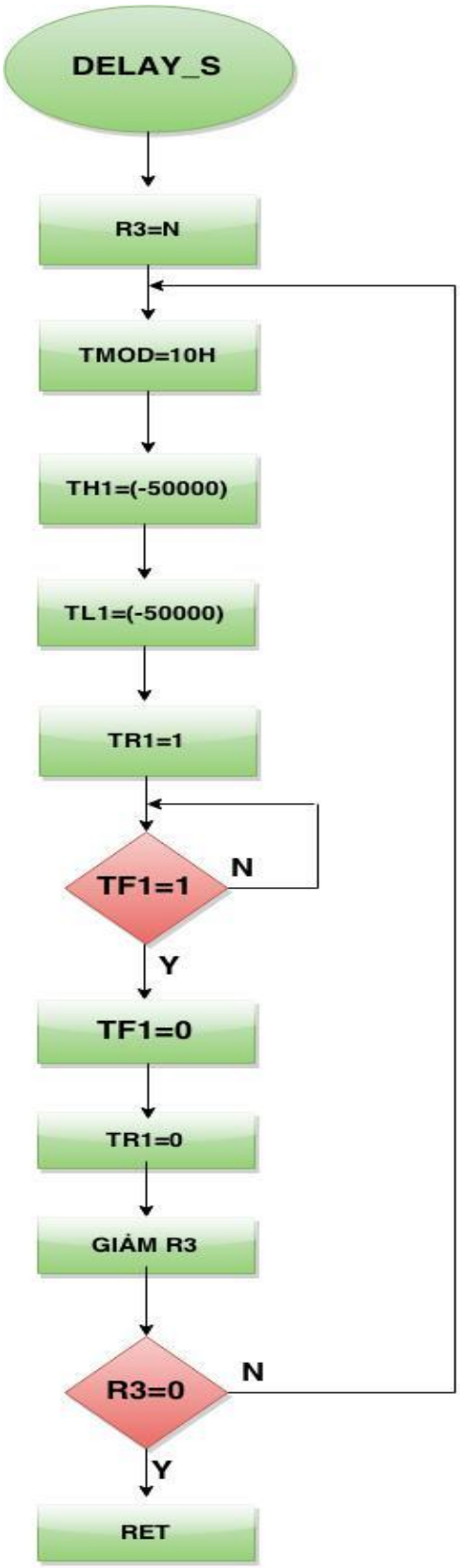
Chương trình :

```
DELAY_MS:
MOV TMOD,#01H
MOV TH0,#HIGH(-1500)
MOV TL0,#LOW(-1500)
SETB TR0
JNB TF0,$
CLR TR0
CLR TF0
RET
```

4.2.2 Hàm delay_s

- Hàm delay_s sử dụng tao thời gian trẻ sau mỗi trạng thái, bằng cách thay đổi giá trị nạp vào R3 cho phù hợp ta được thời gian trể tương ứng là n*50000us hàm delay_s ở đây sử dụng timer1 để tính toán thời gian trể

Thuật toán :


DELAY_S
R3=N
TMOD=10H
TH1=(-50000)
TL1=(-50000)
TR1=1
TF1=1
N
Y
TF1=0
TR1=0
GIẢM R3
R3=0
N
Y
RET

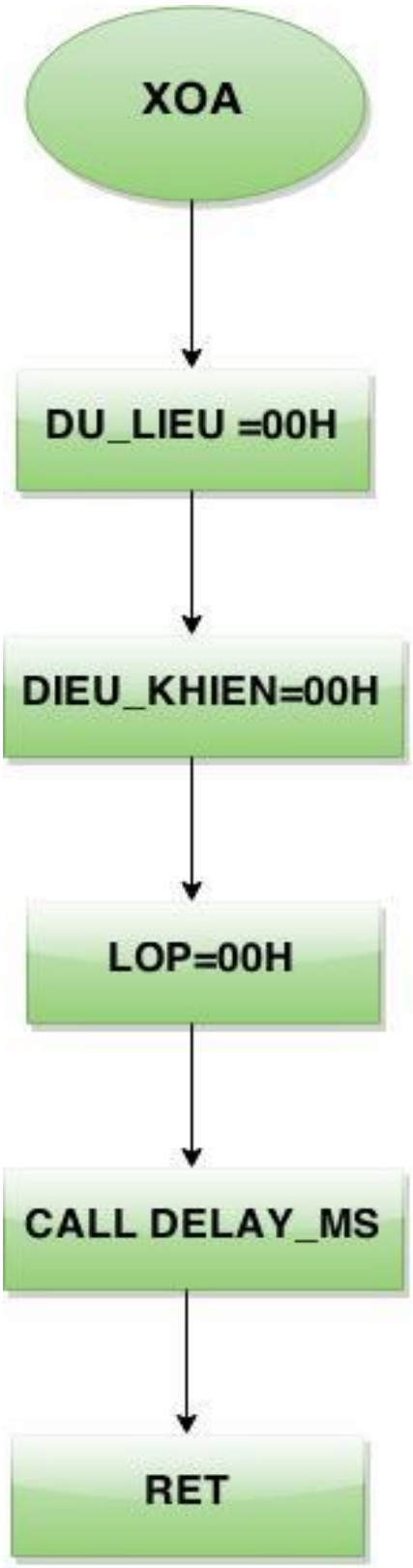
Chương trình :

```
DELAY_S:
MOV R2,#10
DELAY_1:
MOV TMOD,#10H
MOV TH1,#HIGH(-50000)
MOV TL1,#LOW(-50000)
SETB TR1
JNB TF1,$
CLR TF1
CLR TR1
DJNZ R2,DELAY_1
RET
```

4.2.3 Hàm xóa dữ liệu

- Hàm này nhằm mục đích xóa dữ liệu đầu ra để chống nhiễu, chống lém sau mỗi lần quét LED.

Thuật toán :

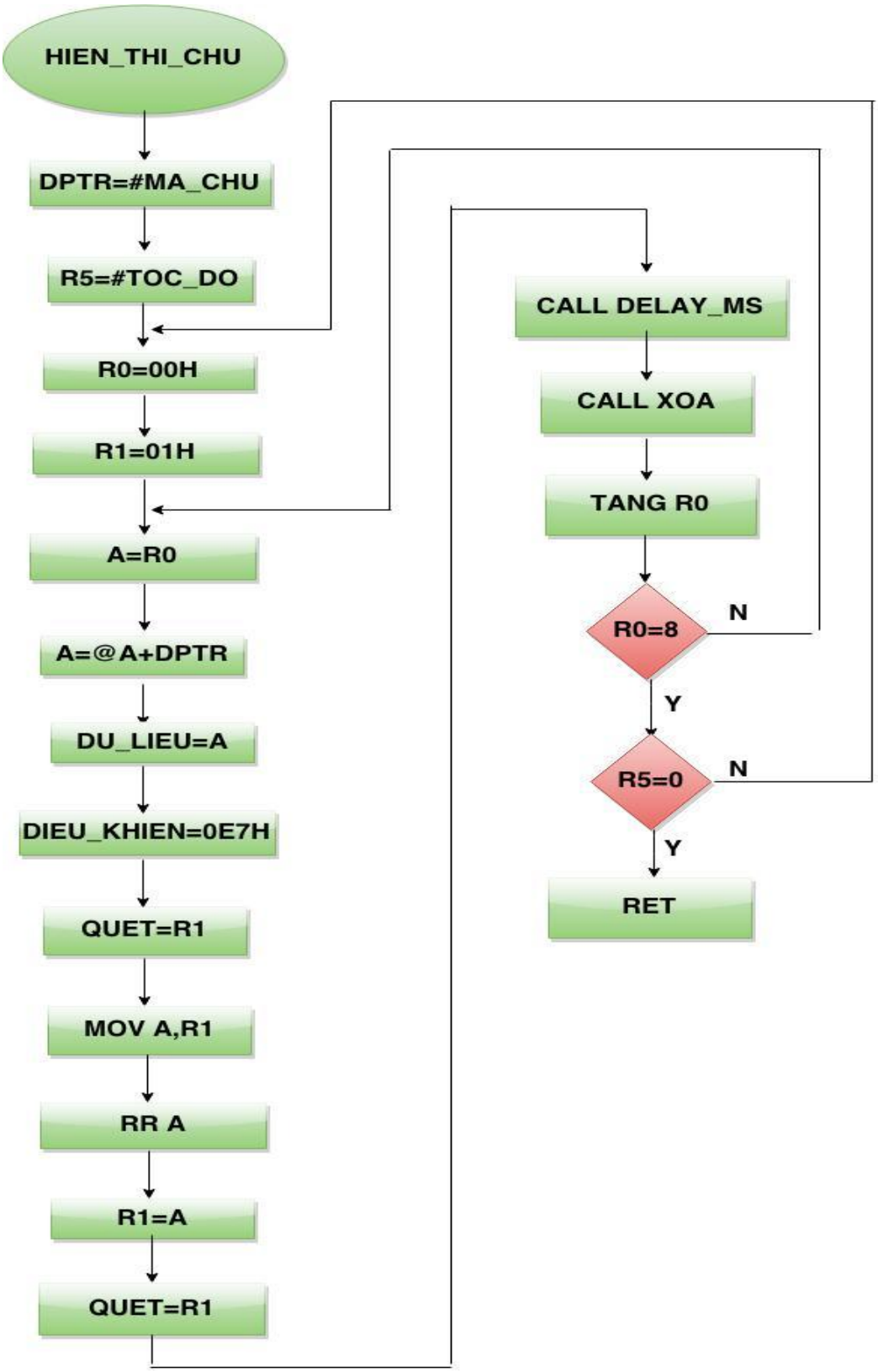

XOA
DU_LIEU =00H
DIEU_KHIEN=00H
LOP=00H
CALL DELAY_MS
RET

Chương trình :

```
;_____HAM XOA CHONG NHIEU_____
XOA:
MOV DU_LIEU,#00H
MOV DIEU_KHIEN,#00H
MOV QUET,#00H
CALL DELAY_MS
RET
```

4.2.4 Hàm hiển thị

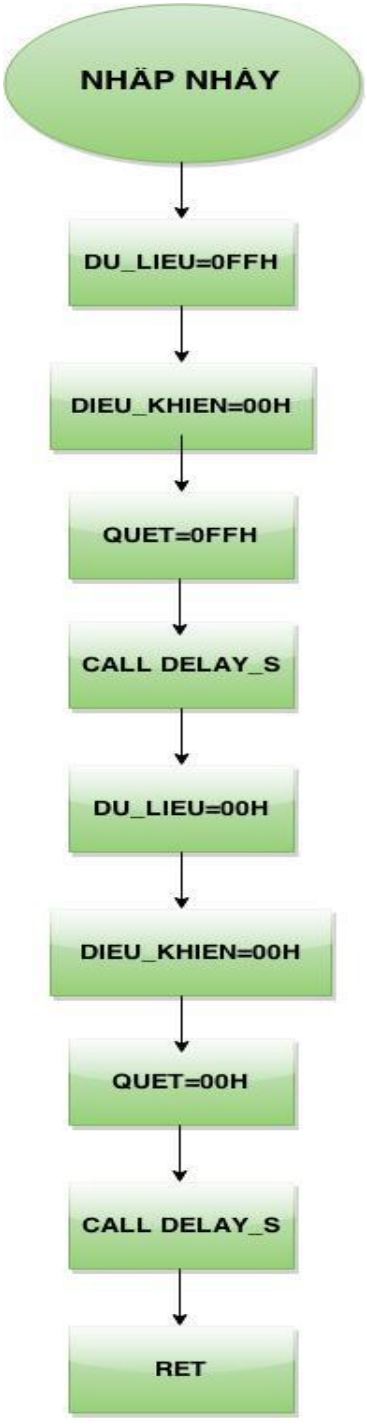
- Hàm này dùng để lấy dữ liệu từ bảng mã quét và hiển thị kí tự, hình ảnh lên khối LED


HIEN_THI_CHU
DPTR=#MA_CHU
R5=#TOC_DO
R0=00H
R1=01H
A=R0
A=@A+DPTR
DU_LIEU=A
DIEU_KHIEN=0E7H
QUET=R1
MOV A,R1
RR A
R1=A
QUET=R1
CALL DELAY_MS
CALL XOA
TANG R0
R0=8
N
Y
R5=0
N
Y
RET

4.3. Thuật toán và chương trình một số hiệu ứng đơn giản

4.3.1 Hiệu ứng khối LED nhấp nháy trong khoảng thời gian delay_s

Thuật toán :

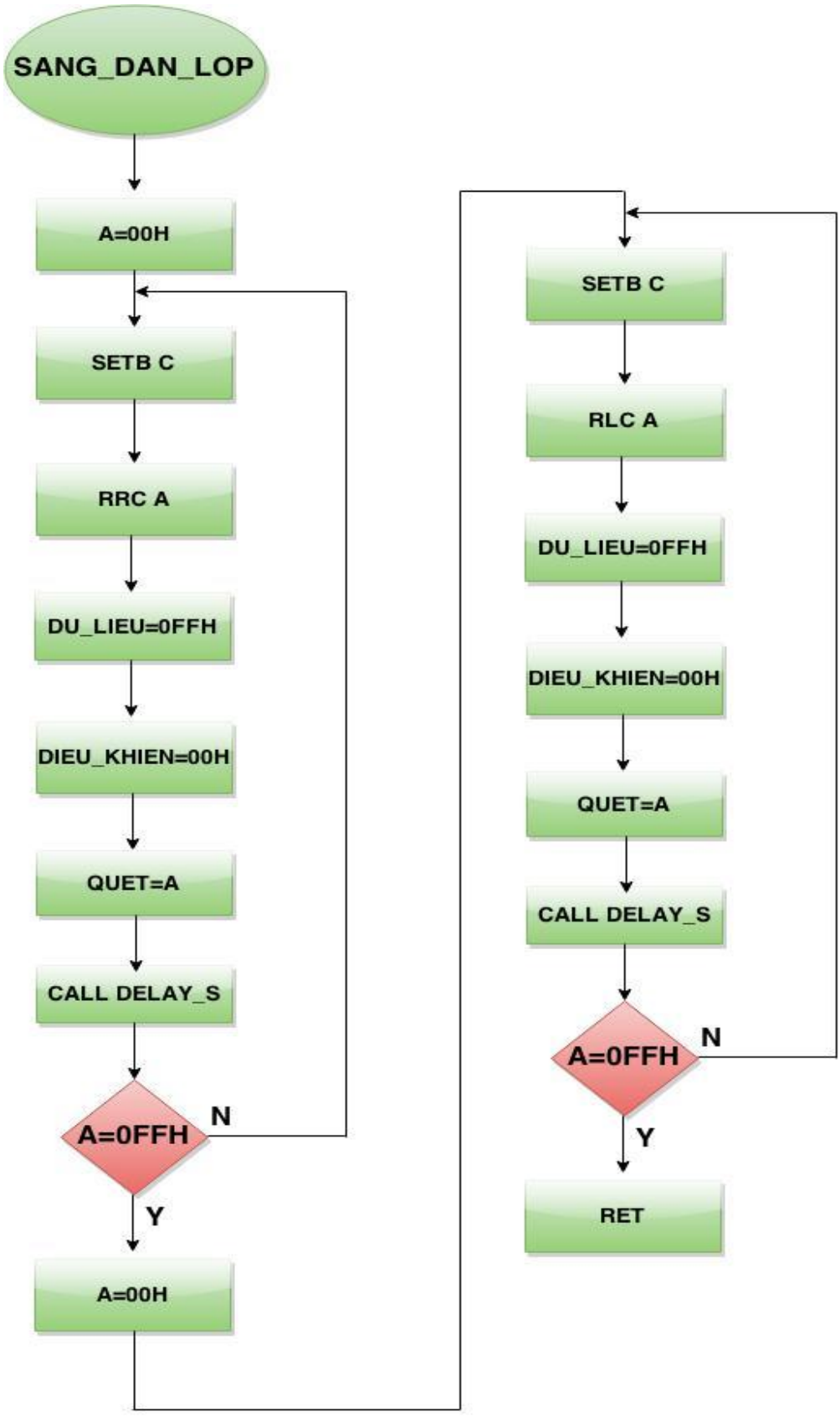


Chương trình :

```
NHAP_NHAY:
MOV R2,#3
LAP_1:
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,#0FFH
CALL DELAY_S
MOV DU_LIEU,#00H
MOV DIEU_KHIEN,#00H
MOV QUET,#00H
CALL DELAY_S
DJNZ R2,LAP_1
RET
```

4.3.2 Hiệu ứng lớp sáng dần từ trên xuống dưới và ngược lại

Thuật toán :

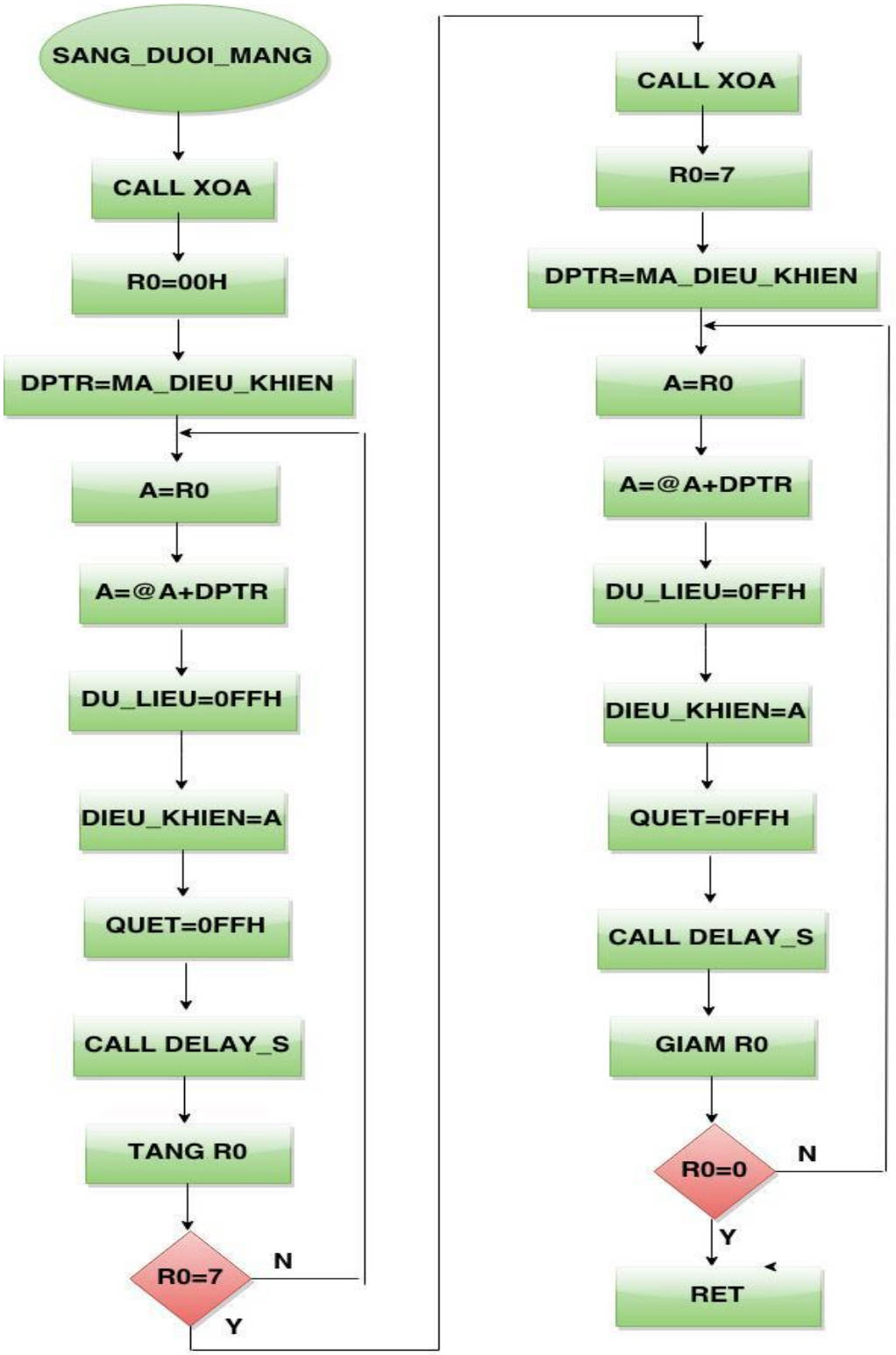

SANG_DAN_LOP
A=00H
SETB C
RRC A
DU_LIEU=0FFH
DIEU_KHIEN=00H
QUET=A
CALL DELAY_S
A=0FFH
N
Y
A=00H
SETB C
RLC A
DU_LIEU=0FFH
DIEU_KHIEN=00H
QUET=A
CALL DELAY_S
A=0FFH
N
Y
RET

Chương trình :

```
SANG_DAN_LOP:
MOV A,#00H
DICH_1:
SETB C
RRC A
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
CALL DELAY_S
CJNE A,#0FFH,DICH_1
;_____________________________________
MOV A,#00H
DICH_2:
SETB C
RLC A
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
CALL DELAY_S
CJNE A,#0FFH,DICH_2
RET
```

4.3.3 Hiệu ứng lớp sáng đuổi từ trên xuống dưới và ngược lại

Thuật toán :



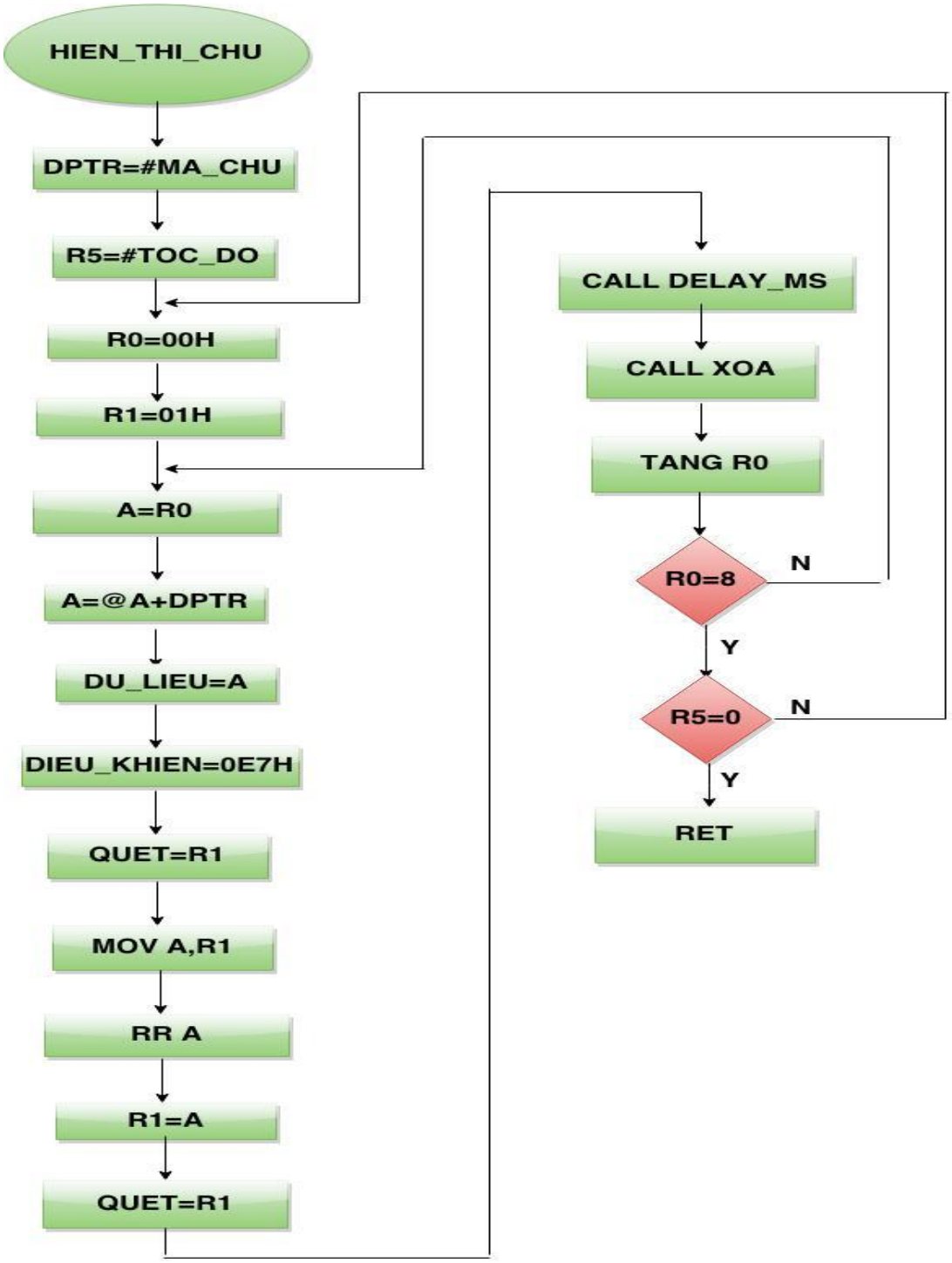
Chương trình :

```
SANG_DUOI_LOP:
MOV R1,#SO_LAN_LAP
LAP_LAI:
CALL XOA
MOV A,#10000000B
LAP_2:
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
RR A
CALL DELAY_S
CJNE A,#10000000B,LAP_2
;_____________________________________________________________
CALL XOA
MOV A,#00000001B
LAP_3:
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
RL A
CALL DELAY_S
CJNE A,#00000001B,LAP_3
DJNZ R1,LAP_LAI
```

RET

4.4 Thuật toán và chương trình một số hiệu ứng khác

4.4.1 Hiện thị chữ giữa các mảng ở dạng tĩnh



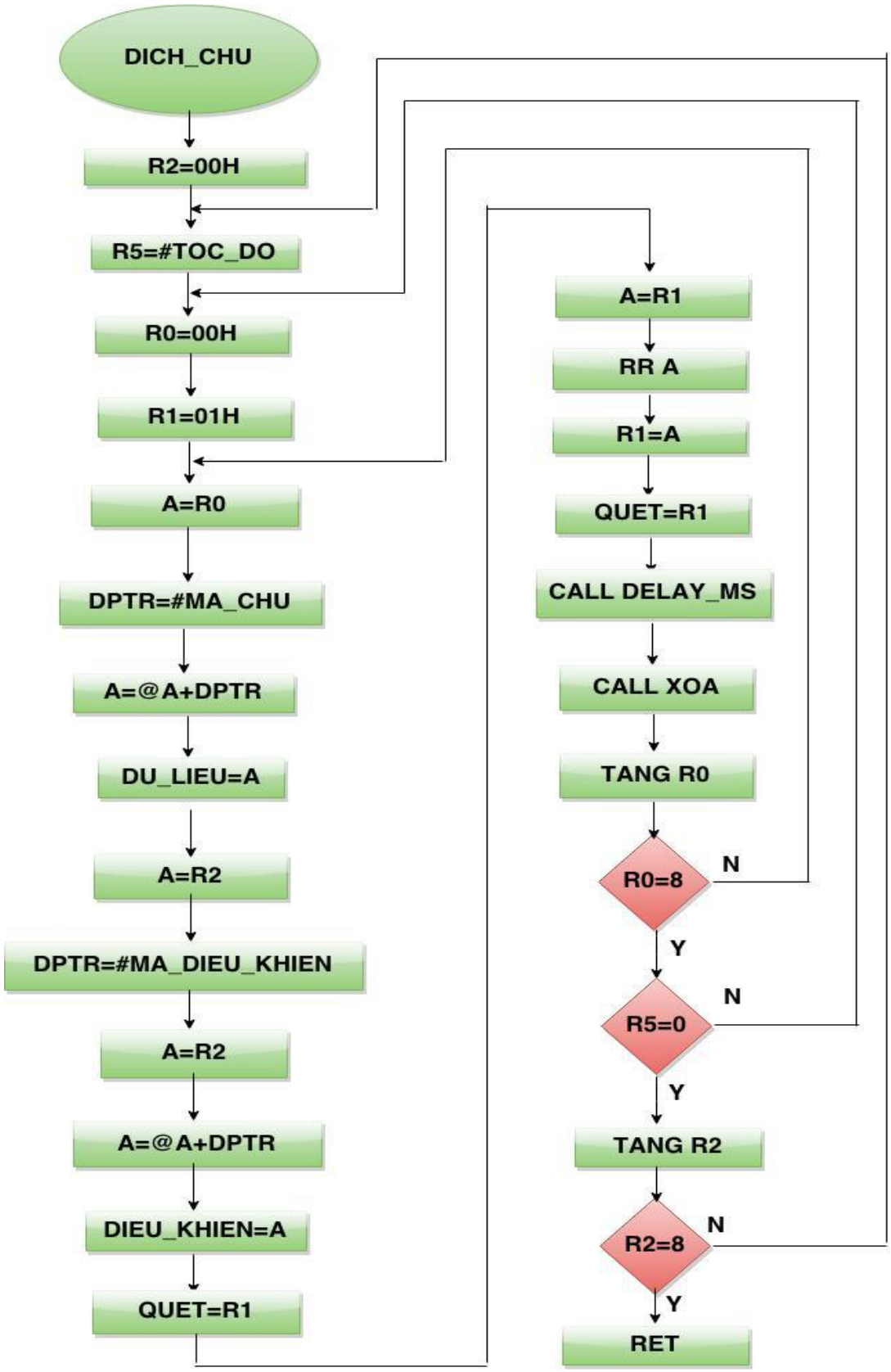
Chương trình :

```
HIEN_CHU_MRTU:
MOV DPTR,#MA_CHU_M
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_R
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_T
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_U
CALL HIEN_THI_CHU
RET
;_____________________________
HIEN_THI_CHU:
MOV R5,#TOC_DO_HIEN_THI
TOC_DO:
MOV R0,#00H
MOV R1,#00000001B
QUET_LED:
MOV A,R0 ;
MOVC A,@A+DPTR ;
MOV DU_LIEU,A  ;
MOV DIEU_KHIEN,#0E7H
;_____________________________________
MOV QUET,R1
```

```
MOV A,R1
RR A
MOV R1,A
MOV QUET,R1
;_____________________________
CALL DELAY_MS
CALL XOA
INC R0
CJNE R0,#8,QUET_LED
DJNZ R5,TOC_DO
RET
MA_CHU_MRTU:  DB 066H,0FFH,0FFH,0DBH,0DBH,0C3H,0C3H,000H;
              DB 01FH,033H,033H,01FH,00FH,01BH,033H,000H
              DB 07EH,018H,018H,018H,018H,018H,018H,000H
              DB 066H,066H,066H,066H,066H,066H,03CH,000H
;_____________________________________________________________________
```

4.4.2 Hiện thị và dịch chữ giữa các mảng

Thuật toán :


DICH_CHU
R2=00H
R5=#TOC_DO
R0=00H
R1=01H
A=R0
DPTR=#MA_CHU
A=@A+DPTR
DU_LIEU=A
A=R2
DPTR=#MA_DIEU_KHIEN
A=R2
A=@A+DPTR
DIEU_KHIEN=A
QUET=R1
A=R1
RR A
R1=A
QUET=R1
CALL DELAY_MS
CALL XOA
TANG R0
R0=8
N
Y
R5=0
N
Y
TANG R2
R2=8
N
Y
RET

Chương trinh :

```
DICH_CHU:
MOV R2,#00H
DICH:
MOV R5,#TOC_DO_HIEN_THI
TOC_DO_DICH:
MOV R0,#00H
MOV R1,#00000001B
QUET_LED_DICH:
MOV A,R0
MOV DPTR,#MA_CHU_M
MOVC A,@A+DPTR ;
MOV DU_LIEU,A  ;
MOV A,R2
MOV DPTR,#MA_DIEU_KHIEN
MOVC A,@A+DPTR ;
MOV DIEU_KHIEN,A
;_____________________________________
MOV QUET,R1
MOV A,R1
RR A
MOV R1,A
MOV QUET,R1
;_____________________________________
```

```
CALL DELAY_MS
CALL XOA
INC R0
CJNE R0,#8,QUET_LED_DICH
DJNZ R5,TOC_DO_DICH
INC R2
CJNE R2,#8,DICH
RET
```

4.5 Toàn bộ chương trình

```
ORG 00H
;DO AN VI DIEU KHIEN LED 3D CUBE 8X8X8
;NGUOI THUC HIEN : NGUYEN TRONG TUAN ANH 11CDT1
;GVHD :GVHD: Th.s LE XUNG
;TRINH BIEN DICH KEIL C
DU_LIEU DATA P1
DIEU_KHIEN DATA P3
QUET DATA P2
SO_LAN_LAP EQU 2
TOC_DO_HIEN_THI EQU 20
;_____CHUONG TRINH CHINH_____
MAIN:
CALL DICH_CHU
```

```
CALL CHAY_CHU_DUOI_LEN
CALL HIEN_CHU_MRTU
CALL SANG_DUOI_MANG
CALL SANG_TRAI_PHAI
CALL SANG_DUOI_LOP
CALL SANG_DAN_LOP
CALL NHAP_NHAY
JMP MAIN
;_____DICH_CHU_____
DICH_CHU:
MOV R2,#00H
DICH:
MOV R5,#TOC_DO_HIEN_THI
TOC_DO_DICH:
MOV R0,#00H
MOV R1,#00000001B
QUET_LED_DICH:
MOV A,R0
MOV DPTR,#MA_CHU_M
MOVC A,@A+DPTR ;
MOV DU_LIEU,A  ;

MOV A,R2
MOV DPTR,#MA_DIEU_KHIEN
```

```
MOVC A,@A+DPTR ;
MOV DIEU_KHIEN,A
;_____________________________________
MOV QUET,R1
MOV A,R1
RR A
MOV R1,A
MOV QUET,R1
;_____________________________________
CALL DELAY_MS
CALL XOA
INC R0
CJNE R0,#8,QUET_LED_DICH
DJNZ R5,TOC_DO_DICH
INC R2
CJNE R2,#8,DICH
RET
;_____________________HIEN CHU MRTU_____________________
HIEN_CHU_MRTU:
MOV DPTR,#MA_CHU_M
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_R
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_T
```

```
CALL HIEN_THI_CHU
MOV DPTR,#MA_CHU_U
CALL HIEN_THI_CHU
RET
;_____________________________
HIEN_THI_CHU:
MOV R5,#TOC_DO_HIEN_THI
TOC_DO:
MOV R0,#00H
MOV R1,#00000001B
QUET_LED:
MOV A,R0 ;
MOVC A,@A+DPTR ;
MOV DU_LIEU,A  ;
MOV DIEU_KHIEN,#0E7H
;_____________________________________________________
MOV QUET,R1
MOV A,R1
RR A
MOV R1,A
MOV QUET,R1
;_____________________________________________________
CALL DELAY_MS
CALL XOA
```

```
INC R0
CJNE R0,#8,QUET_LED
DJNZ R5,TOC_DO
RET
;_____ _____CHAY CHU_____
CHAY_CHU_DUOI_LEN:
MOV DPTR,#MA_CHU_MRTU
MOV R7,#32
CHAY_CHU:
MOV R5,#5
TOC_DO2:
MOV R0,#00H
MOV R1,#00000001B
QUET_LED2:
MOV A,R0
MOVC A,@A+DPTR
MOV DU_LIEU,A
MOV DIEU_KHIEN,#0E7H
;_____
MOV QUET,R1
MOV A,R1
RR A
MOV R1,A
MOV QUET,R1
```

```
;_____________________________________________________________________
CALL DELAY_MS
CALL XOA
INC R0
CJNE R0,#8,QUET_LED2
DJNZ R5,TOC_DO2
INC DPTR
DJNZ R7,CHAY_CHU
RET
;_____________SANG DUOI MANG_____________________
SANG_DUOI_MANG:
CALL XOA
MOV R0,#00H
MOV DPTR,#MA_DIEU_KHIEN
LABEL_1:
MOV A,R0
MOVC A,@A+DPTR
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,A
MOV QUET,#0FFH
CALL DELAY_S
INC R0
CJNE R0,#7,LABEL_1
;/////////////////
```

```
CALL XOA
MOV R0,#7
;Nap di chi vung du lieu
LABEL_2:
MOV A,R0
MOVC A,@A+DPTR
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,A
MOV QUET,#0FFH
CALL DELAY_S
DEC R0
CJNE R0,#0,LABEL_2
RET
;_____SANG_MANG_____
SANG_TRAI_PHAI:
MOV R1,#3
LAP_TP:
CALL XOA
MOV R0,#00H
MOV DPTR,#MA_TRAI_PHAI
LABEL_TP1:
MOV A,R0
MOVC A,@A+DPTR
MOV DU_LIEU,A
```

```
MOV DIEU_KHIEN,#00H
MOV QUET,#0FFH
CALL DELAY_S
INC R0
CJNE R0,#7,LABEL_1
;_____________________________________________
CALL XOA
MOV R0,#7
;//MOV DPTR,#MA_MANG
LABEL_TP2:
MOV A,R0
MOVC A,@A+DPTR
MOV DU_LIEU,A
MOV DIEU_KHIEN,#00H
MOV QUET,#0FFH
CALL DELAY_S
DEC R0
CJNE R0,#0,LABEL_2
DJNZ R1,LAP_TP
RET
;_____________________SANG_DUOI_LOP_____________________________
SANG_DUOI_LOP:
MOV R1,#SO_LAN_LAP
LAP_LAI:
```

```
CALL XOA
MOV A,#10000000B
LAP_2:
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
RR A
CALL DELAY_S
CJNE A,#10000000B,LAP_2
;_____________________________________________________________
CALL XOA
MOV A,#00000001B
LAP_3:
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
RL A
CALL DELAY_S
CJNE A,#00000001B,LAP_3
DJNZ R1,LAP_LAI
RET

;_____________________SANG DAN LOP_____________________________
SANG_DAN_LOP:
```

```
MOV A,#00H
DICH_1:
SETB C
RRC A
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
CALL DELAY_S
CJNE A,#0FFH,DICH_1
;_____________________________________
MOV A,#00H
DICH_2:
SETB C
RLC A
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,A
CALL DELAY_S
CJNE A,#0FFH,DICH_2
RET
;_____________NHAP NHAY_____________________
NHAP_NHAY:
MOV R2,#3
LAP_1:
```

```
MOV DU_LIEU,#0FFH
MOV DIEU_KHIEN,#00H
MOV QUET,#0FFH
CALL DELAY_S
MOV DU_LIEU,#00H
MOV DIEU_KHIEN,#00H
MOV QUET,#00H
CALL DELAY_S
DJNZ R2,LAP_1
RET
;_____HAM XOA CHONG NHIEU_____
XOA:
MOV DU_LIEU,#00H
MOV DIEU_KHIEN,#00H
MOV QUET,#00H
CALL DELAY_MS
RET
;/_____HAM DELAY_MS_____
DELAY_MS:
MOV TMOD,#01H
MOV TH0,#HIGH(-1500)
MOV TL0,#LOW(-1500)
SETB TR0
JNB TF0,$
```

```
CLR TR0
CLR TF0
RET
;_____HAM DELAY_S_____
DELAY_S:
MOV R2,#10
DELAY_1:
MOV TMOD,#10H
MOV TH1,#HIGH(-50000)
MOV TL1,#LOW(-50000)
SETB TR1
JNB TF1,$
CLR TF1
CLR TR1
DJNZ R2,DELAY_1
RET
;_____BANG MA_____
MA_TRAI_PHAI: DB 001H,002H,004H,008H,010H,020H,040H,080H
MA_DIEU_KHIEN: DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
MA_CHU_M:    DB 066H,0FFH,0FFH,0DBH,0DBH,0C3H,0C3H,000H
MA_CHU_R:    DB 01FH,033H,033H,01FH,00FH,01BH,033H,000H
MA_CHU_T:    DB 07EH,018H,018H,018H,018H,018H,018H,000H
MA_CHU_U:    DB 066H,066H,066H,066H,066H,066H,03CH,000H
```

```
MA_CHU_MRTU:    DB 066H,0FFH,0FFH,0DBH,0DBH,0C3H,0C3H,000H;
                DB 01FH,033H,033H,01FH,00FH,01BH,033H,000H
                DB 07EH,018H,018H,018H,018H,018H,018H,000H
                DB 066H,066H,066H,066H,066H,066H,03CH,000H
END
;_____MRTU_____;
```

CHƯƠNG 5

KẾT LUẬN–TÀI LIỆU, PHẦN MỀM THAM KHẢO

5.1. Hạn chế và hướng phát triển

- Do giới hạn của đồ án là lập trình bằng ngôn ngữ ASM, nên giới hạn về thuật toán, hàm hổ trợ nên ở đây em chỉ trình bày theo phương pháp thông dụng đó là xuất dữ liệu, quét và hiển thị. Do đó chỉ hiển thị được những hiệu ứng không quá phức tạp, khó có thể áp dụng theo phương pháp tọa độ để lập trình để tạo nên nhưng hiệu ứng phức tạp đẹp mắt hơn nữa. Và một phần do điểm yếu của vi điều khiển 8051 tốc độ chậm so với những vi điều khiển thông dụng hiện nay, bộ nhớ RAM ít việc truy xuất dữ liệu chậm khó có thể áp dụng các hàm toán học vào chương trình.

- Để giải quyết vấn đề này chúng ta có thể thay thế bằng những vi điều khiển mạnh hơn đó là PIC, AVR, MSP430, hày là ARM ….. Ngoải ra ta có thể phát triển thêm khối LED có thêm chức năng giao tiếp với máy tính thông qua cổng COM, USB để truyền dữ liệu từ máy tính xuống tốt hơn giúp tối ưu được thuật toán, ít tốt bộ nhớ RAM, và việc thay đổi hiệu ứng, hình ảnh đơn giản, dễ dàng hơn.

5.2.Tài liệu và phần mềm sử dụng

Tài liệu tham khảo :

- Cấu trúc và lập trình họ vi điều khiển 8051 (Nguyễn Tăng Cường- Phan Quốc Thắng)

- Họ vi dieu khien 8051 (Tống Văn On)

Phần mềm sử dụng :

- Trình biên dịch KEIL C4

- Trình biên dịch MSC51-IDE

- Phần mềm tạo mã LED: LED MATRIX V1.2

- Phần mềm tạo mã LED cube: Gamo LED cube
- Phần mềm vẽ mạch ORCAD

- Phần mêm mô phỏng PROTEUS

Trang web tham khảo :

- Mrtu.blogspot.com

- Codientu.org.com
- Dientuvietnam.net
- Google.com
- Facebook.com
- Youtube.com