**II**

|  |
| --- |
| **ĐẠI HỌC BÁCH KHOA HÀ NỘI**  TRƯỜNG ĐIỆN - ĐIỆN TỬ |

**---🙠**🕮**🙢---**

****

**BÁO CÁO ĐỒ ÁN I**

**ĐỀ TÀI:**

**Cảm biến đo khoảng cách sử dụng vi xử lý**

|  |  |
| --- | --- |
| **GV HƯỚNG DẪN:** | **THS.Nguyễn Thị Huế** |
| **SINH VIÊN** | **Đinh Nhật Hoàng 20191850**  **Lê Thanh Hải 20191813** |
|  |  |

***Hà Nội, 08/2022***

**Mục Lục**

[LỜI NÓI ĐẦU 2](#_Toc111283547)

[I. Giới thiệu về các thiết bị được sử dụng 3](#_Toc111283548)

[*1. Cảm biến siêu âm* 3](#_Toc111283549)

[*1.1* *Giới thiệu về nguyên lý* 4](#_Toc111283550)

[*1.2 Hoạt động phát và nhận phản hồi song âm cơ bản của SRF05* 6](#_Toc111283551)

[*1.3 Một số ứng dụng của SRF05 và bộ vi điều khiển* 8](#_Toc111283552)

[*2. Giới thiệu về vi xử lý họ 8051* 9](#_Toc111283553)

[*2.1 Tổ chức bộ nhớ* 9](#_Toc111283554)

[*2.2 Các thanh ghi đặc biệt* 11](#_Toc111283555)

[*2.3 Lập trình cho vi điều khiển 89S52* 14](#_Toc111283556)

[3. Các thiết bị bổ sung 14](#_Toc111283557)

[*3.1 LCD1602* 14](#_Toc111283558)

[*2.Biến trở, điện trở, tụ điện, dây nguồn, cổng chuyển đổi rs232 to usb.* 16](#_Toc111283559)

[II. Thiết kế mạch nguyên lý, mô phỏng và xây dựng mạch PCB 16](#_Toc111283560)

[*1.* *Sơ đồ khối* 16](#_Toc111283561)

[*2.* *Mạch nguyên lý* 17](#_Toc111283562)

[*3.* *Mạch mô phỏng* 17](#_Toc111283563)

[*4.* *Mạch thiết kế* 18](#_Toc111283564)

[*5.* *Chương trình cho mạch* 19](#_Toc111283565)

[*5.1 Lưu đồ thuật toán* 19](#_Toc111283566)

[*5.2 Chương trình cho vi xử lý* 19](#_Toc111283567)

[III. KẾT LUẬN 26](#_Toc111283568)

[*1. Hình ảnh sản phẩm* 26](#_Toc111283569)

[*2. Thử nghiệm sản phẩm* 26](#_Toc111283570)

[*3. Kết luận và hướng phát triển.* 27](#_Toc111283571)

# LỜI NÓI ĐẦU

Ngày nay, trong các hệ thống đo lường - điều khiển, mọi quá trình đều được đặc trưng bởi các biến trạng thái. Các biến trạng thái này thường là các đại lượng không điện như nhiệt độ, áp suất, lưu lượng, tốc độ, độ di chuyển v.v…

Để thực hiện các quá trình đo lường và điều khiển cần phải thu thập thông tin, đo đạc, theo dõi sự biến thiên của các biến trạng thái của quá trình thực hiện chức năng trên là các thiết bị cảm biến.

Cảm biến là các phần tử nhạy cảm dùng để biến đổi các đại lượng đo lường, kiểm tra hay điều khiển từ dạng này sang dạng khác thuận tiện hơn cho việc tác động của các phần tử khác. Cảm biến là một thiết bị chịu tác động của đại lượng cần đo m không có tính chất điện và cho một đặc trưng mang bản chất điện (như điện tích, điện áp, dòng điện, trở kháng) kí hiệu là s có: s = F(m). Cảm biến thường dùng ở khâu đo lường và kiểm tra.

Các loại cảm biến được sử dụng rộng rãi trong tự động hóa các quá trình sản xuất và điều khiển tự động các hệ thống khác nhau. Chúng có chức năng biến đổi sự thay đổi liên tục các đại lượng đầu vào .

Căn cứ theo dạng đại lượng đầu vào người ta phân ra các loại cảm biến như: cảm biến chuyển dịch thẳng, chuyển dịch góc quay, tốc độ, gia tốc, momen quay, nhiệt độ, áp suất, quang, bức xạ v.v...

Các thiết bị cảm biến đang dần trở thành một phần không thể thiếu trong đời sống hiện đại của chúng ta. Trong đề tài này này, chúng em chỉ xét đến cảm biến đo khoảng cách. Do kiến thức còn hạn chế và thời gian tìm hiểu chưa được nhiều nên đề tài này này còn nhiều thiếu xót. Chúng em hi vọng sẽ nhận được nhiều ý kiến từ giáo viên hướng dẫn để được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

**I. Giới thiệu về các thiết bị được sử dụng**

1. Cảm biến siêu âm



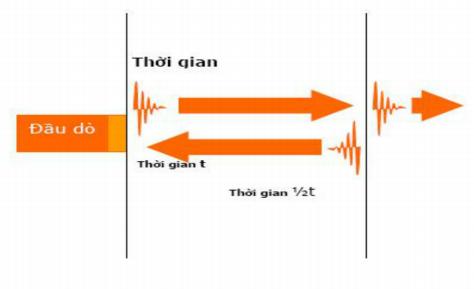
I.1.1 Một số cảm biến siêu âm

Cảm biến siêu âm có nhiều loại, tùy thuộc theo công dụng như để nhận biết vật trong khoảng cách gần hay xa, nhận biết các vật có tính chất khác nhau và trong những điều kiện hoạt động khác nhau mà người ta chế tạo các loại cảm biến siêu âm cũng khác nhau.

* 1. Giới thiệu về nguyên lý

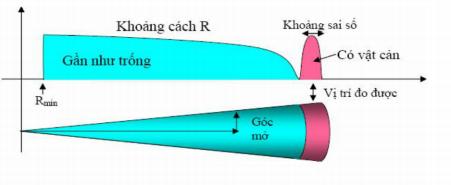
**a.** **Cảm biến siêu âm và nguyên tắc TOF (** Time Of Flight )

Quãng đường di chuyển của sóng sẽ bằng 2 lần khoảng cách từ cảm biến tới chướng ngại vật sẽ được tính theo nguyên lý TOF:  d = v\* t/2



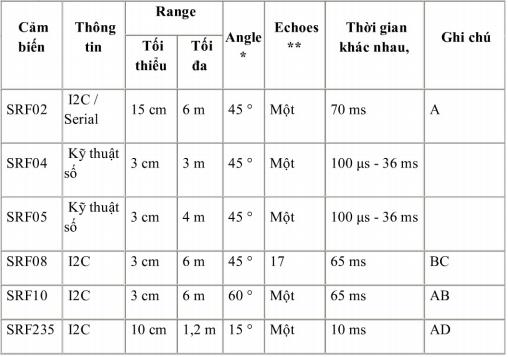
I.1.2 Nguyên lý làm việc cảm biến siêu âm

**b.** **Tầm quét của cảm biến siêu âm**



I.1.3 Tầm quét

**c.** **Thông số một số loại cảm biến siêu âm SRF**



\*: Ước tính góc của hình nón cảm biến ở ½ cảm biến

\*\*: Số vọng ghi lại bởi cảm biến. Đây là những tiếng vọng ghi từ đọc gần đây nhất, và được ghi đè mới bằng mỗi lần khác nhau.

A: Những cảm biến nhỏ hơn điển hình ( SRF05/04) kích thước.

B: Phạm vi thời gian có thể được điều chỉnh xuống bằng cách điều chỉnh được.

C: Cảm biến này cũng bao gồm một photocell ở mặt trước để phát hiện ánh sáng.

D: Hoạt động ở một tần số 235kHz cao hơn.

**d.** **Cảm biến siêu âm SRF05 và đặc tính kỹ thuật**

**SRF05** là một bước phát triển từ SRF04, được thiết kế làm tăng tính linh hoạt, tăng phạm vi, ngoài ra còn giảm bớt chi phí. SRF05 là hoàn toàn tương thích với SRF04. Khoảng cách được tăng từ 3 – 4m.

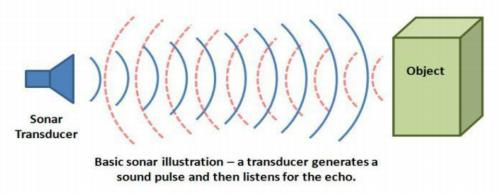


I.1.4 SRF05

**SRF05** cho phép sử dụng một chân duy nhất cho cả kích hoạt và phản hồi, do đó tiết kiệm giá trị trên chân điều khiển. Khi chân chế độ không kết nối, thì SRF05 hoạt động riêng biệt chân kích hoạt và chân hồi tiếp, như SRF04. SRF05 bao gồm một thời gian trễ trước khi xung phản hồi để mang lại điều khiển chậm hơn hẳn như bộ điều khiển thời gian cơ bản Stamps và Picaxe để thực hiện các xung lệnh.

1.2 Hoạt động phát và nhận phản hồi song âm cơ bản của SRF05

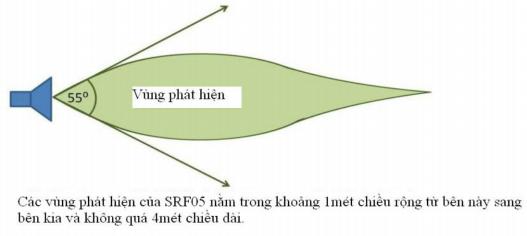
**Nguyên tắc cơ bản của sonar:** là tạo ra một xung âm thanh điện tử và sau đó lắng nghe tiếng vọng tạo ra khi các làn sóng âm thanh số truy cập một đối tượng và được phản xạ trở lại. Để tính thời gian cho phản hồi trở về, một ước tính chính xác có thể được làm bằng khoảng cách tới đối tượng. Xung âm thanh tạo ra bởi SRF05 là siêu âm, nghĩa là nó ở trên phạm vi nhận xét của con người. Trong khi tần số thấp hơn có thể được sử dụng trong các loại ứng dụng, tần số cao hơn thực hiện tốt hơn cho phạm vi ngắn, nhu cầu độ chính xác cao.



I.1.5 Nguyên lý sonar

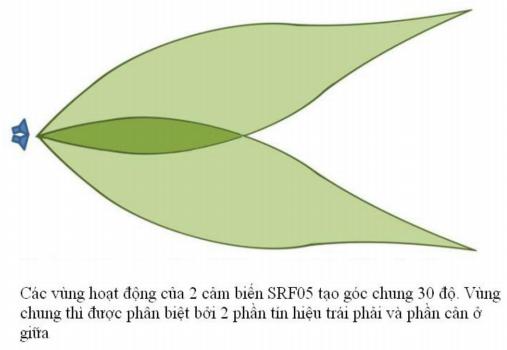
**Vùng phát hiện của SRF05**

* Nếu ngưỡng để phát hiện đối tượng được đặt quá gần với cảm biến, các đối tượng trên một đường có thể bị va chạm tại một điểm mù. Nếu ngưỡng này được đặt ở một khoảng cách quá lớn từ các cảm biến thì đối tượng sẽ được phát hiện mà không phải là trên một đường va chạm.



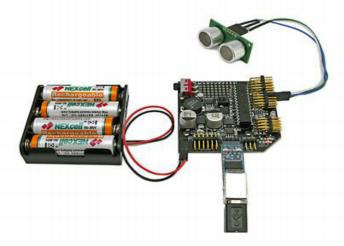
I.1.6 Vùng phát hiện

* Một kỹ thuật phổ biến để làm giảm các điểm mù và đạt được phát hiện chiều rộng lớn hơn cự ly gần là thêm một cải tiến bằng cách thêm một đơn vị SRF05 bổ sung và gắn kết của hai đơn vị hướng về phía trước. Thiết lập như vậy thì có một khu vực mà hai khu vực phát hiện chồng chéo lên nhau.



I.1.7 Vùng hoạt động

1.3 Một số ứng dụng của SRF05 và bộ vi điều khiển



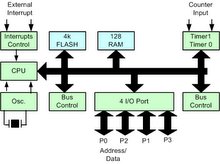
I.1.8 Ứng dụng của cảm biến

2. Giới thiệu về vi xử lý họ 8051

2.1 Tổ chức bộ nhớ

    Các vi điều khiển thuộc họ 8051 đều tổ chức thành 2 không gian chương trình và dữ liệu, hình 1.1.1 và hình 1.1.2 sẽ mô tả điều này. Kiến trúc vi xử lý 8 bit của 8051 này cho phép truy nhập và tính toán nhanh hơn đối với không gian dữ liệu nhờ việc phân chia 2 không gian bộ nhớ chương trình và dữ liệu như trên. Tuy nhiên bộ nhớ ngoài được truy nhập bởi hệ thống 16 bit địa chỉ vẫn có thể thực hiện nhờ thanh ghi con trỏ.

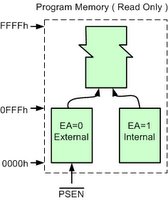
    Bộ nhớ chương trình (ROM, EPROM) là bộ nhớ chỉ đọc, có thể mở rộng tối đa 64Kbyte. Vói họ vi điều khiển 89xx, bộ nhớ chương trình được tích hợp sẵn trong chip có kích thước nhỏ nhất là 4kByte. Với các vi điều khiển không tích hợp sẵn bộ nhớ chương trình trên chip, buộc phải thiết kế bộ nhớ chương trình bên ngoài. Ví dụ sử dụng EPROM: 2764 (64Kbyte), khi đó chân PSEN phải ở mức tích cực (5V).

[](http://machdientu.googlepages.com/intro1.gif/intro1-full.jpg)

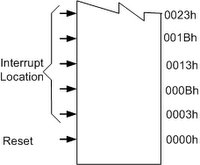
Hình I.2.1 Cấu trúc vi điều khiển 89C51

     Bộ nhớ dữ liệu (RAM) tồn tại độc lập so với bộ nhớ chương trình. Họ vi điều khiển 8051 có bộ nhớ dữ liệu tích hợp trên chip nhỏ nhất là 128byte và có thể mở rộng với bộ nhớ dữ liệu ngoài lên tới 64kByte. Với những vi điều khiển không tích hợp ROM trên chip thì vẫn có RAM trên chip là 128byte. Khi sử dụng RAM ngoài, CPU đọc và ghi dữ liệu nhờ tín hiệu trên các chân RD và WR. Khi sử dụng cả bộ nhớ chương trình và bộ nhớ dữ liệu bên ngoài thì buộc phải kết hợp chân RD và PSEN bởi cổng logic AND để phân biệt tín hiệu truy xuất dữ liệu trên ROM hay RAM ngoài.

*Bộ nhớ chương trình:*

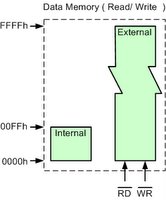
[](http://machdientu.googlepages.com/intro2.gif/intro2-full.jpg)

 Hình I.2.2 Cấu trúc bộ nhớ chương trình

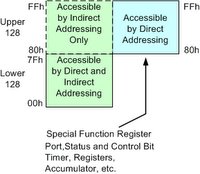
[](http://machdientu.googlepages.com/intro4.gif/intro4-full;init:.jpg)

Hình I.2.3 Địa chỉ các ngắt trên bộ nhớ chương trình

*Bộ nhớ dữ liệu:*

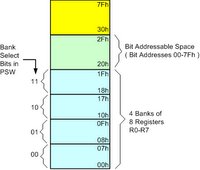
[](http://machdientu.googlepages.com/intro3.gif/intro3-full.jpg)

Hình I.2.4 Cấu trúc bộ nhớ dữ liệu

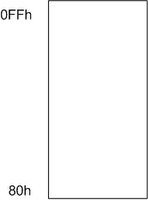
[](http://machdientu.googlepages.com/intro5.gif/intro5-full.jpg)

 Hình I.2.5 Cấu trúc bộ nhớ trong

   .

[](http://machdientu.googlepages.com/intro6.gif/intro6-full.jpg)

 Hình I.2.6 Cấu trúc 128 byte thấp của bộ nhớ dữ liệu trong

[](http://machdientu.googlepages.com/intro7.gif/intro7-full.jpg)

 Hình I.2.7. 128 byte cao của bộ nhớ dữ liệu.

2.2 Các thanh ghi đặc biệt

*Thanh ghi chính*   Thanh ghi tính toán  chính của vi điều khiển 8051 ACC (Accumulator). Là thanh ghi đặc biệt của 8051 dùng để thực hiện các phép toán của CPU, thường kí hiệu là A.

*Thanh ghi phụ*

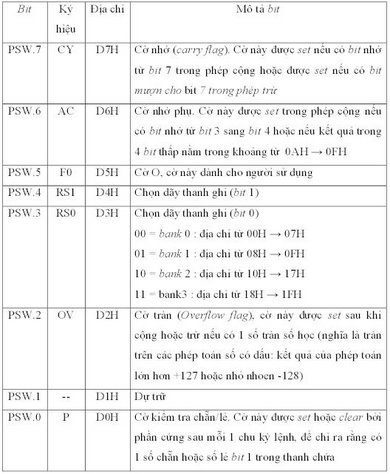
    Thanh ghi tính toán phụ của vi điều khiển 8051 là B. Thanh ghi B ở địa chỉ F0H được dùng chung với thanh chứa A trong các phép toán nhân, chia.

*Thanh ghi trạng thái chương trình (PSW)*

.

[Description: 398,48](http://machdientu.googlepages.com/ThanhghiPSW1.jpg/ThanhghiPSW1-full.jpg)

Hình I.2.8 Thanh ghi trạng thái chương trình PSW

[](http://machdientu.googlepages.com/ThanhghiPSW2.jpg/ThanhghiPSW2-full.jpg)

 Hình I.2.9 Chi tiết các bit trong thanh ghi PSW

*Thanh ghi ngăn xếp (Stack Pointer)*

    Con trỏ stack SP (stack pointer) là 1 thanh ghi 8 bit ở địa chỉ 81H. SP chứa địa chỉ của dữ liệu hiện đang ở đỉnh của stack. Các lệnh liên quan đến satck bao gồm lệnh cất dữ liệu vào stack và lệnh lấy dữ liệu ra khỏi stack. Việc cất vào stack làm tăng SP trước khi ghi dữ liệu và việc lấy dữ liệu ra khỏi stack sẽ giảm SP. Vùng stack của 8051 được giữ trong RAM nội  và được giới hạn đến các địa chỉ truy xuất được bởi kiểu định địa chỉ gián tiếp. Các lệnh PUSH và POP sẽ cất dữ liệu vào stack và lấy dữ liệu từ stack, các lệnh gọi chương trình con (ACALL, LCALL) và lệnh trở về (RET, RETI) cũng cất và phục hồi nội dung của bộ đếm chương trình PC (Program counter)

*Con trỏ dữ liệu DPTR*

     Con trỏ dữ liệu DPTR (data pointer) được dùng để truy xuất bộ nhớ chương trình ngoài hoặc bộ nhớ dữ liệu ngoài. DPTR là một thanh ghi 16 bit có địa chỉ là 82H (DPL, byte thấp) và 83H (DPH, byte cao).

*Thanh ghi các cổng P0-P3*

    Các port xuất/nhập của 8051 bao gồm  Port 0 tại địa chỉ 80H, Port 1 tại địa chỉ 90H, Port 2 tại địa chỉ A0H và Port 3 tại địa chỉ B0H. Tất cả các port đều được định địa chỉ từng bit nhằm cung cấp các khả năng giao tiếp mạnh.

*Thanh ghi bộ đệm truyền thông nối tiếp (Serial Data Buffer)*

    Bộ đệm truyền thông được chia thành hai bộ đệm, bộ đệm truyền dữ liệu và bộ đệm nhận dữ liệu. Khi dữ liệu được chuyển vào thanh ghi SBUF, dữ liệu sẽ được chuyển vào bộ đệm truyền dữ liệu và sẽ được lưu giữ ở đó cho đến khi quá trình truyền dữ liệu qua truyền thông nối tiếp kết thúc. Khi thực hiện việc chuyển dữ liệu từ SBUF ra ngoài, dữ liệu sẽ được lấy từ bộ đệm nhận dữ liệu của truyền thông nối tiếp.

*Thanh ghi của bộ định thời/bộ đếm*

    8051 có 2 bộ đếm/định thời (counter/timer) 16 bit để định các khoảng thời gian hoặc để đếm các sự kiện. Các cặp thanh ghi (TH0, TL0) và (TH1, TL1) là các thanh ghi của bộ đếm thời gian. Bộ định thời 0 có địa chỉ 8AH (TL0, byte thấp) và 8CH (TH0, byte cao). Bộ định thời 1 có địa chỉ 8BH (TL1, byte thấp) và 8DH (TH1, byte cao).

    Hoạt động của bộ định thời được thiết lập bởi thanh ghi chế độ định thời TMOD (Timer Mode Register) ở địa chỉ 88H. Chỉ có TCON được định địa chỉ từng bit.

*Các thanh ghi điều khiển*

    Các thanh ghi điều khiển đặc biệt như IP, IE, TMOD, TCON, SCON và PCON là các thanh ghi điều khiển và ghi nhận trạng thái của hệ thống ngắt, bộ đếm/định thời, truyền thông nối tiếp. Chi tiết của các thanh ghi này sẽ được mô tả sau.

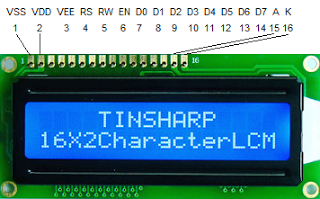
2.3 Lập trình cho vi điều khiển 89S52

Chúng ta hay sử dụng hai ngôn ngữ chính là C và assembly. Nhưng trong đồ án này chúng em chọn ngôn ngữ C để lập trình. Ngôn ngữ C có tính tùy biến cao và dễ quản lý trong các dự án lớn.

Trình dịch sử dụng là Keil C do hãng Atmel khuyên dùng.

## 3. Các thiết bị bổ sung

3.1 LCD1602

[](http://www.dientumaytinh.com/2012/03/gioi-thieu-co-ban-ve-lcd-16x2.html)

*Hình I.3.1 Sơ đồ chân của LCD*

a. Chức năng các chân :

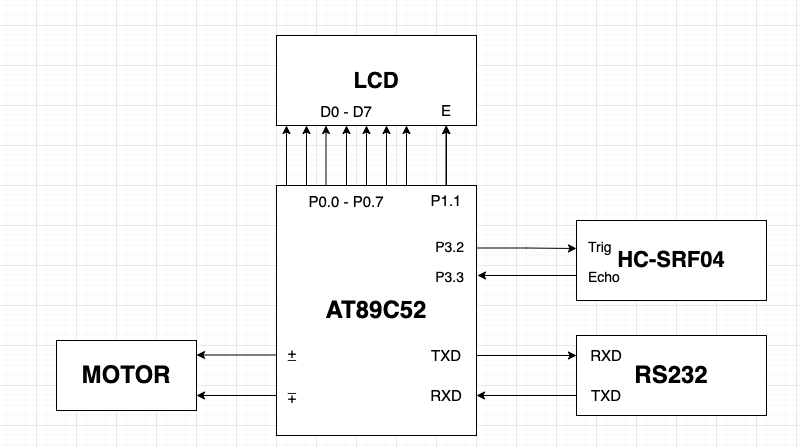
|  |  |  |
| --- | --- | --- |
| Chân | Ký hiệu | Mô tả |
| 1 | Vss | Chân nối đất cho LCD, khi thiết kế mạch ta nối chân này với GND của mạch điều khiển |
| 2 | VDD | Chân cấp nguồn cho LCD, khi thiết kế mạch ta nối chân này với VCC=5V của mạch điều khiển |
| 3 | VEE | Điều chỉnh độ tương phản của LCD. |
| 4 | RS | Chân chọn thanh ghi (Register select). Nối chân RS với logic “0” (GND) hoặc logic “1” (VCC) để chọn thanh ghi.  + Logic “0”: Bus DB0-DB7 sẽ nối với thanh ghi lệnh IR của LCD (ở chế độ “ghi” - write) hoặc nối với bộ đếm địa chỉ của LCD (ở chế độ “đọc” - read)  + Logic “1”: Bus DB0-DB7 sẽ nối với thanh ghi dữ liệu DR bên trong LCD. |
| 5 | R/W | Chân chọn chế độ đọc/ghi (Read/Write). Nối chân R/W với logic “0” để LCD hoạt động ở chế độ ghi, hoặc nối với logic “1” để LCD ở chế độ đọc. |
| 6 | E | Chân cho phép (Enable). Sau khi các tín hiệu được đặt lên bus DB0-DB7, các lệnh chỉ được chấp nhận khi có 1 xung cho phép của chân E.  + Ở chế độ ghi: Dữ liệu ở bus sẽ được LCD chuyển vào(chấp nhận) thanh ghi bên trong nó khi phát hiện một xung (high-to-low transition) của tín hiệu chân E.  + Ở chế độ đọc: Dữ liệu sẽ được LCD xuất ra DB0-DB7 khi phát hiện cạnh lên (low-to-high transition) ở chân E và được LCD giữ ở bus đến khi nào chân E xuống mức thấp. |
| 7 - 14 | DB0 - DB7 | Tám đường của bus dữ liệu dùng để trao đổi thông tin với MPU. Có 2 chế độ sử dụng 8 đường bus này :  + Chế độ 8 bit : Dữ liệu được truyền trên cả 8 đường, với bit MSB là bit DB7.  + Chế độ 4 bit : Dữ liệu được truyền trên 4 đường từ DB4 tới DB7, bit MSB là DB7 |
| 15 | - | Nguồn dương cho đèn nền |
| 16 | - | GND cho đèn nền |

*Bảng 1 : Chức năng các chân của LCD*

2.Biến trở, điện trở, tụ điện, dây nguồn, cổng chuyển đổi rs232 to usb.

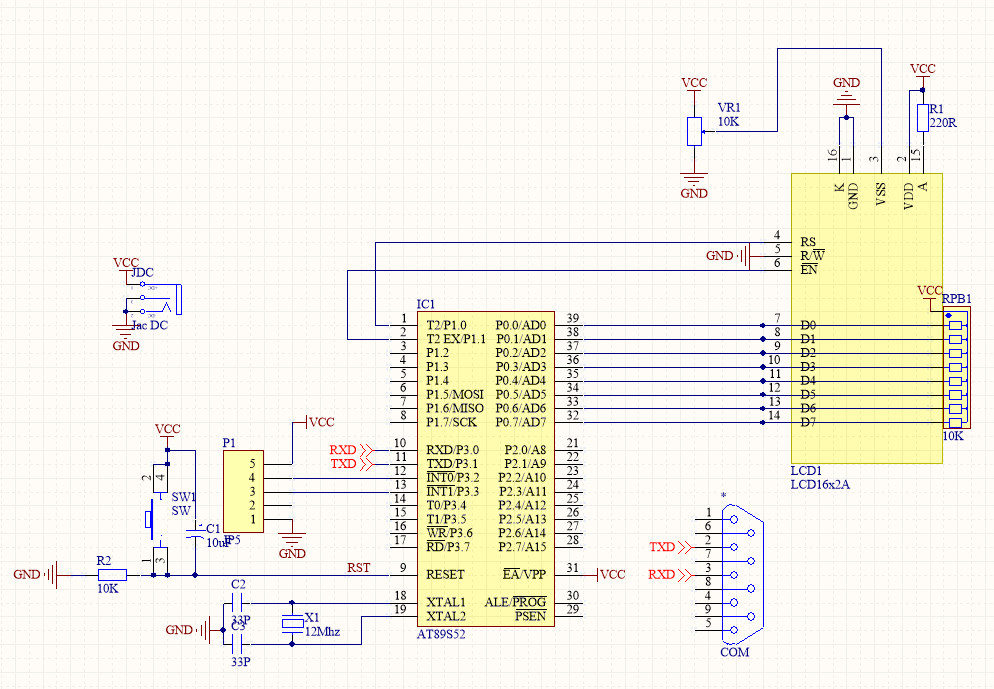
**II. Thiết kế mạch nguyên lý, mô phỏng và xây dựng mạch PCB**

* 1. Sơ đồ khối



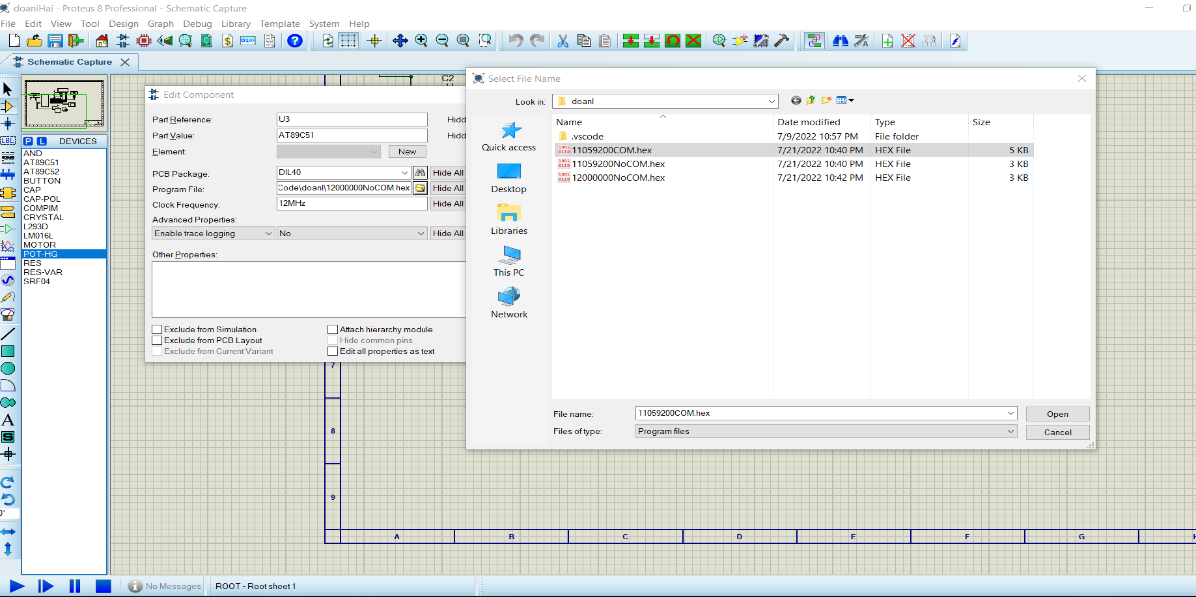
Cảm biến phát xung sau đó truyền tín hiệu về vi xử lý.Vi xử lý có nhiệm vụ phân tích dữ liệu, hiển thị lên LCD và truyền thông qua cổng kết nối RS232. Khi muốn thay đổi khoảng cách, ta nhập khoảng cách mong muốn trên màn hình. Thông qua RS232, vi xử lí nhận thông tin và điều khiển motor quay cho đến khi cần.

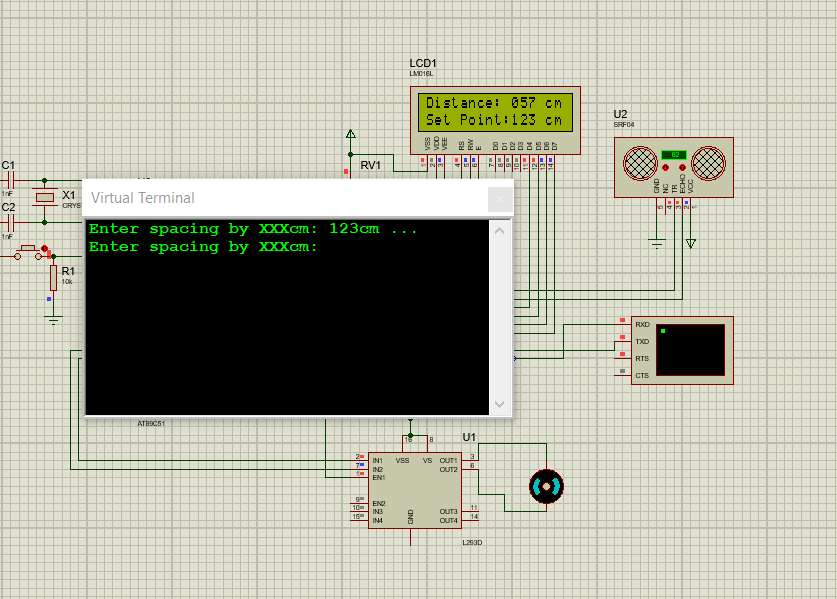
* 1. Mạch nguyên lý



* 1. Mạch mô phỏng

Sau khi thiết kế được mạch nguyên lý thì chúng em sẽ sử dụng proteus để mô phỏng cho mạch.



Nhập số liệu và chạy thử

* 1. Mạch thiết kế

A picture containing text

Description automatically generated

* 1. Chương trình cho mạch

5.1 Lưu đồ thuật toán

Diagram

Description automatically generated

*5.2 Chương trình cho vi xử lý*

Chương trình được viết bằng ngôn ngữ C trên trình dịch keil C

#include "AT89X52.h"

sbit LCD\_EN = P1^1;

sbit LCD\_RS = P1^0;

sbit TRIG = P3^2;

sbit ECHO = P3^3;

sbit Am = P1^2;

sbit Duong = P1^3;

void Uart\_Write(char c)

{

    while(TI == 0);

    TI = 0;

    SBUF = c;

}

void Uart\_Write\_Text(char \* str)

{

    unsigned char i = 0;

    while(str[i]!=0)

    {

        Uart\_Write(str[i]);

        i++;

    }

}

void delay\_ms(unsigned int x) // 12MHZbit

    {

    unsigned int i,j;

    for(i=0;i<=x;i++)

    {

        for(j=0;j<=110;j++);

    }

}

void delay\_us(unsigned char x)

{

        x=x/10;

        while(x--);

}

void chot\_dl()

{

    LCD\_EN =1;

    delay\_us(3);

    LCD\_EN=0;

    delay\_us(50);

}

void lcd\_cmd (unsigned char command )

{

    P0=command;

    chot\_dl();

}

void lcd\_putchar( char dat)

{

    LCD\_RS=1;

    lcd\_cmd(dat);

    LCD\_RS=0 ;

}

void lcd\_puts (char \*s)

{

    while (\*s)

    {

        lcd\_putchar(\*s);

        s++;

    }

}

void lcd\_gotoxy(unsigned char x, unsigned char y)

{

    unsigned char address;

    if(y==0)

    address = (0x80+x);

    else

    address = (0xc0+x);

    delay\_us(1000);

    lcd\_cmd(address);

    delay\_us(50);

}

void lcd\_init()

{

    LCD\_RS=0;

    lcd\_cmd( 0x38 );

    lcd\_cmd( 0x0c);

    lcd\_cmd( 0x06 );

    lcd\_cmd( 0x01 );

}

void in\_so(unsigned int num)

{

    char a,b,c;

    a=num/100;

    b=(num%100)/10;

    c=(num%100)%10;

    lcd\_putchar(a+48);

    lcd\_putchar(b+48);

    lcd\_putchar(c+48);

}

void init\_timer()

{

    TH1=TL1=0;

    TR1=0;

}

void timer\_off(){ TR1=0; }

void timer\_on(){ TR1=1; TH1=TL1=0;}

unsigned int get\_peroid()

{

    TRIG=1;

    delay\_us(12);

    TRIG=0;

    while(ECHO==0);

    timer\_on();

    while(ECHO==1);

    timer\_off();

    return TH1\*256 + TL1;

}

unsigned char setPoint = 50;

unsigned long int testSetPoint = 50;

unsigned char setPoints[5];

unsigned char distance = 0;

unsigned int i = 0;

void main()

{

    TMOD = 0x10;

    SM0 = 0;

    SM1 = 1;

    TH1 = 0xFD;             // 9600

    TR1 = 1;                // Timer1 bat dau chay

    TI = 1;                 // San sang gui du lieu

    REN = 1;                // Cho phep nhan du lieu

    lcd\_init();

    delay\_us(1000);

    init\_timer();

    while(1)

    {

        lcd\_gotoxy(0,0);

        lcd\_puts("Distance: ");

        distance = get\_peroid()/57;

        in\_so(distance);

        lcd\_puts("Cm");

        lcd\_gotoxy(0,1);

        lcd\_puts("Set Point: ");

        in\_so(setPoint);

        lcd\_puts("Cm");

        if(RI == 1) {

            RI = 0;

            setPoints[i] = SBUF - 128;

            if(setPoints[i] >= 0x30 && setPoints[i] <= 0x39) {

                Uart\_Write(setPoints[i]);

                i++;

            }

            if(i == 3) {

                testSetPoint = (setPoints[0] - 0x30) \* 100 + (setPoints[1] - 0x30) \* 10 + (setPoints[2] - 0x30);

                i = 0;

                Uart\_Write\_Text("cm ...");

                Uart\_Write(0x0D);

                if(testSetPoint < 255) {

                    setPoint = testSetPoint;

                } else {

                    Uart\_Write\_Text("Set point invalid!");

                    Uart\_Write(0x0D);

                }

                Uart\_Write\_Text("Enter spacing by XXXcm: ");

            }

        }

        if (distance > setPoint) {

            Am = 1;

            Duong = 0;

        }

        if (distance < setPoint) {

            Am = 0;

            Duong = 1;

        }

        delay\_ms(100);

    }

**III. KẾT LUẬN**

1. Hình ảnh sản phẩm

A picture containing text

Description automatically generated

III.1.1 Hình ảnh sản phẩm

2. Thử nghiệm sản phẩm

Thí nghiệm đo với một khoảng cách bất kì:

Ví dụ: Khoảng cách 8cm

A picture containing wall, indoor

Description automatically generated

III.2.1 Kết quả thử nghiệm

3. Kết luận và hướng phát triển.

Đề tài này chúng em có thể phát triển để đo mức chất lỏng, robot tự hành tránh vật cản hay bất kì một ứng dụng nào liên quan tới đo khoảng cách.

Sau khi làm đồ án chúng em đã tích lũy được nhiều kinh nghiệm như lập trình vi điều khiển, làm mạch. Tuy nhiên không tránh được những sai sót mong thầy cô trong hội đồng góp ý để chúng em hoàn thiện hơn.

Chúng em xin chân thành cảm ơn.