BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM

KHOA: ĐIỆN – ĐIỆN TỬ

------oOo-----

HCMUTE

BÁO CÁO

Đề tài: Khảo sát chức năng

Hồng ngoại trong Kit IC8051

TÊN SV: Nguyễn Minh Tiến Đạt
MÃ SINH VIÊN: 21139075
LỚP: 21139A
HỌC KỲ 2023 - 2024

TPHCM, Tháng 11

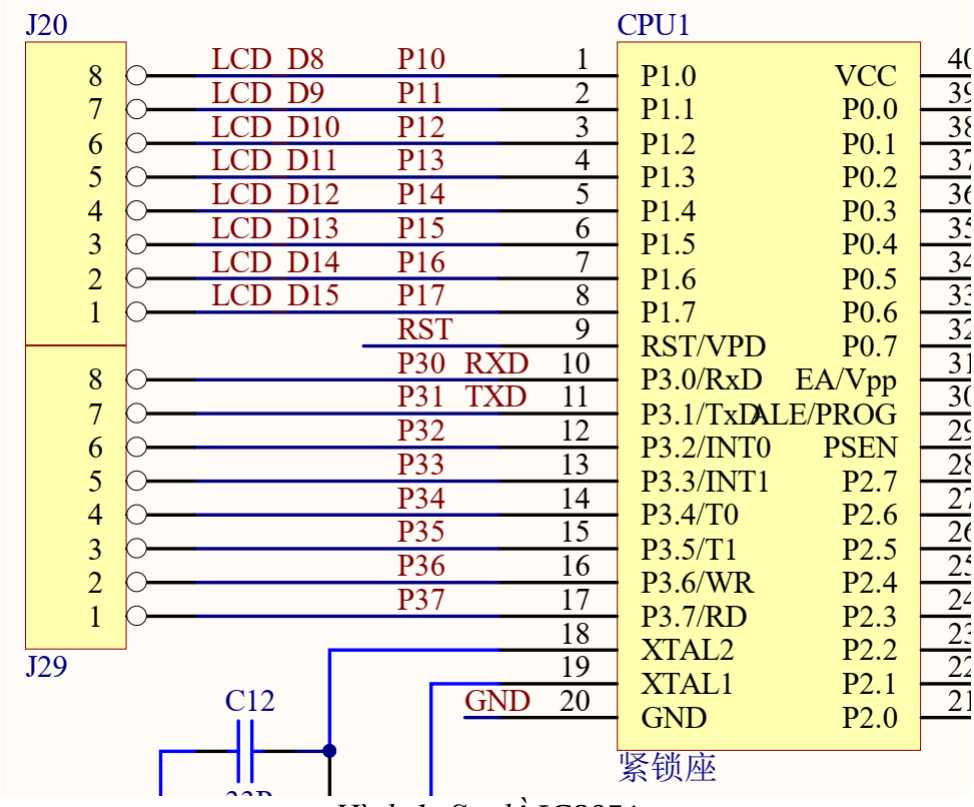
Mục lục

Nội dung

I. Sơ đồ kết nối của chức năng hồng ngoại ở Kit IC8051:

a. Sơ đồ ở IC8051

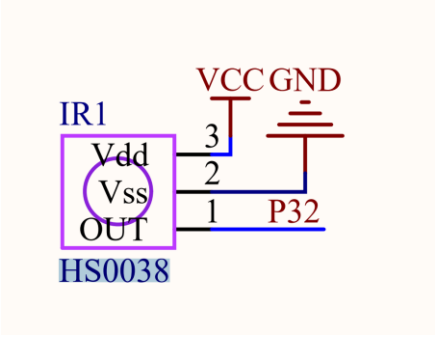
Trong IC8051 chân nhận kết nối của hồng ngoại là chân P32 tức là chân ngắt INT0, điều này quan trọng vì việc biết được chân nhận tín hiệu sẽ là bước quan trọng để hiểu được code ở phần sau



Hình 1: Sơ đồ IC8051

b. Sơ đồ ở IC HS0038

Ở đây chân P32 được kết nối với IC HS0038 là IC nhận tín hiệu hồng ngoại chính của Kit IC8051.



Hình 2: IC HS0038

II. IC sử dụng trong chức năng hồng ngoại ở Kit IC8051:

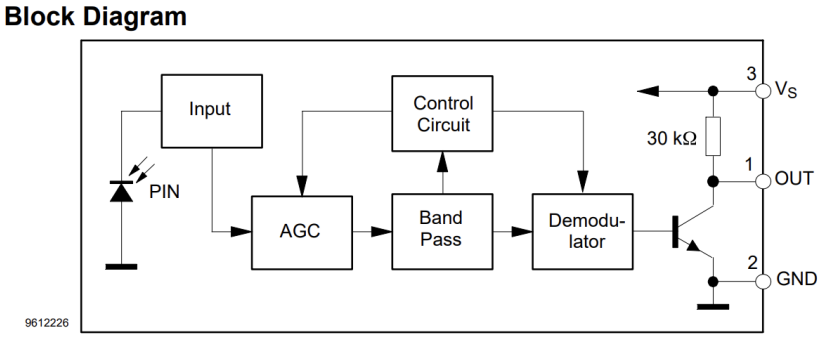
a. Tổng quan về HS0038:

HS0038 là một cảm biến hồng ngoại (IR) phổ biến được sử dụng trong các ứng dụng liên quan đến điều khiển từ xa và truyền thông hồng ngoại. Nó được sử dụng để nhận diện và nhận tín hiệu hồng ngoại từ các bộ điều khiển từ xa, máy tính cá nhân, máy tính bảng và các thiết bị điện tử khác.

HS0038 thường được sử dụng trong các ứng dụng như điều khiển từ xa, hệ thống bảo mật hồng ngoại, các hệ thống tự động hóa và robot. Với khả năng nhận diện tín hiệu hồng ngoại chính xác và đáng tin cậy, nó là một thành phần quan trọng trong việc truyền thông hồng ngoại và điều khiển từ xa.

Hình 3: IC HS0038

b. Chức năng, cấu tạo:



Hình 4: Sơ đồ khối

HS0038 có một mắt nhìn hẹp chuyên dụng để nhận tín hiệu hồng ngoại trong một phạm vi tần số 38 kHz. Điều này làm cho nó tương thích với hầu hết các bộ điều khiển từ xa sử dụng tần số này. Nó có khả năng nhận diện tín hiệu hồng ngoại và chuyển đổi chúng thành tín hiệu điện để xử lý bởi các vi mạch điện tử.

Cảm biến HS0038 thường có ba chân: nguồn cung cấp (Vcc), chân đất (GND) và chân dữ liệu (OUT). Nguồn cung cấp thường là 5V, trong khi chân dữ liệu sẽ xuất ra tín hiệu điện tương ứng với tín hiệu

hồng ngoại nhận được. Điều này cho phép việc kết nối dễ dàng với vi mạch điều khiển như Arduino hoặc vi điều khiển khác.

c. Cách nhận tín hiệu:

Các bước nhận tín hiệu từ Remote đến HS0038:

✓ Remote (bộ điều khiển từ xa) phát tín hiệu hồng ngoại: Remote phát ra một tín hiệu hồng ngoại

✓ Cảm biến HS0038 nhận tín hiệu hồng ngoại: HS0038 có một mắt nhìn hẹp chuyên dụng để nhận tín hiệu hồng ngoại. Khi remote phát tín hiệu, HS0038 nhận tín hiệu hồng ngoại thông qua mắt nhìn này.

✓ Chuyển đổi tín hiệu hồng ngoại thành tín hiệu điện: HS0038 chuyển đổi tín hiệu hồng ngoại nhận được thành tín hiệu điện tương ứng. Khi tín hiệu hồng ngoại có, HS0038 sẽ đưa ra tín hiệu điện cao (high), và khi không có tín hiệu hồng ngoại, nó sẽ đưa ra tín hiệu điện thấp (low).

✓ Đọc tín hiệu điện bằng vi mạch điều khiển: Tín hiệu điện từ HS0038 được đọc bởi vi mạch điều khiển như Arduino hoặc vi điều khiển khác. Vi mạch điều khiển có thể sử dụng chức năng như digitalRead() để đọc giá trị tín hiệu từ chân kết nối với HS0038.

✓ Xử lý tín hiệu điện: Sau khi đọc tín hiệu điện, vi mạch điều khiển có thể xử lý nó để giải mã các lệnh từ remote. Cụ thể, nó có thể so sánh mẫu tín hiệu với các mẫu đã biết để xác định lệnh mà remote đã gửi.

III. Giao thức truyền thông tin sử dụng trong chức năng hồng ngoại:

a. Các giao thức chính:

Trong việc truyền dữ liệu hồng ngoại, có một số giao thức chính được sử dụng. Dưới đây là một số giao thức phổ biến trong truyền dữ liệu hồng ngoại:

- ✓ NEC (Nippon Electric Company) Protocol: NEC Protocol là một giao thức phổ biến được sử dụng trong các ứng dụng điều khiển từ xa. Giao thức này sử dụng tần số 38 kHz và mã hóa thông tin dựa trên độ dài xung và khoảng cách giữa các xung. Mỗi lệnh được mã hóa thành một chuỗi bit, gồm các mã khóa, mã dữ liệu và mã phụ. **Đây cũng chính là giao thức được sử dụng trong code lần này.**
- ✓ Sony SIRC (Sony Infrared Remote Control) Protocol: Giao thức Sony SIRC được sử dụng rộng rãi trong các thiết bị điều khiển từ xa của Sony. Giao thức này sử dụng tần số 40 kHz và mã hóa thông tin dựa trên thời gian xung cao và thời gian xung thấp của tín hiệu hồng ngoại. Mỗi lệnh được mã hóa thành một chuỗi bit, trong đó có các mã khóa, mã dữ liệu và mã phụ.
- ✓ RC-5 (Remote Control 5) Protocol: Giao thức RC-5 là một giao thức tiêu chuẩn được sử dụng trong các ứng dụng điều khiển từ xa. Giao thức này sử dụng tần số 36 kHz và mã hóa thông tin dựa trên thời gian xung cao và thời gian xung thấp của tín hiệu hồng ngoại. Mỗi lệnh được mã hóa thành một chuỗi bit, trong đó có các mã khóa và mã dữ liệu.
- ✓ Philips RC-6 Protocol: Giao thức Philips RC-6 là một giao thức

phổ biến được sử dụng trong các ứng dụng điều khiển từ xa của Philips. Giao thức này sử dụng tần số 36 kHz và mã hóa thông tin dựa trên thời gian xung cao và thời gian xung thấp của tín hiệu hồng ngoại. Mỗi lệnh được mã hóa thành một chuỗi bit, trong đó có các mã khóa, mã dữ liệu và mã phụ.

b. Giao thức được sử dụng trong Kit 8051:

Giao thức được sử dụng trong code mẫu là giao thức NEC.

Gói tin truyền đi của giao thức NEC bao gồm một chuỗi bit để đại diện cho lệnh điều khiển từ xa. Dưới đây là cấu trúc chung của gói tin NEC:

Mã khóa (Address):

- Độ dài: 8 bit
- Mã khóa xác định thiết bị nhận lệnh. Mỗi thiết bị có một mã khóa riêng biệt để phân biệt nó với các thiết bị khác trong cùng một mạng lưới.
- Thông thường, 1-2 bit đầu tiên của mã khóa được sử dụng để xác định loại thiết bị, trong khi các bit còn lại xác định địa chỉ của thiết bị đó.

Mã dữ liệu (Command):

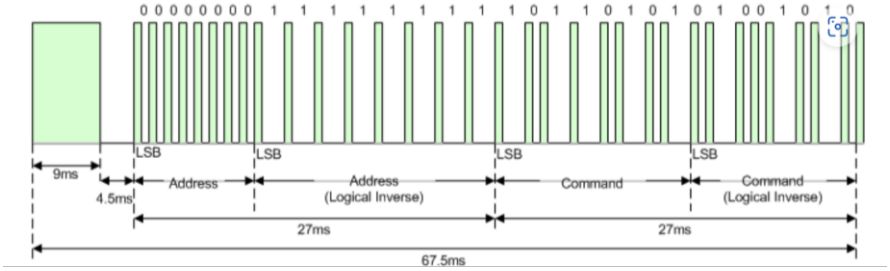
- Độ dài: 8 bit
- Mã dữ liệu biểu thị lệnh cụ thể mà người dùng muốn gửi đến thiết bị nhận lệnh. Ví dụ: tăng/giảm âm lượng, chuyển kênh, tắt/mở nguồn, vv.

Mã phụ (Extended):

- Độ dài: 8 bit
- Mã phụ được sử dụng cho các chức năng mở rộng hoặc điều khiển nâng cao của các thiết bị. Thông thường, mã phụ không được sử dụng trong các ứng dụng cơ bản.

Bit kiểm tra (Checksum):

- Độ dài: 8 bit
- Bit kiểm tra (checksum) được sử dụng để kiểm tra tính toàn vẹn của gói tin. Nó được tính toán từ mã khóa, mã dữ liệu và mã phụ.



Hình 5: Một byte gửi đi của giao thức NEC

IV. Giải thích code mẫu:

a. Sơ bộ khảo sát sau khi test:

Sau khi kết nối và bấm từng nút thì với mỗi nút, màn hình sẽ hiển thi 1 giá trị Hex tương ứng đoạn mã mà remote đã gửi đến IC HS0038, cụ thể như sau:

```
Mode1:
0:16H
1:0C
2:18
3:5E
4:08
5:1C
6:5A
7:42
8:52
9:4A
EQ:07
Nguồn: 45
Mode: 46H
Tatam:47
Next Next: 89 44
Next:87 43
Back: 40
Vol-:15
Vol+:9
rpt: 19
u/sd: 0D
```

Hình 6: Các mã HEX nhận được

Điểm qua một số hàm phụ trong chương trình:

1. Hàm Delay tạo delay khoảng 10.8us

```
57  void DelayMs(unsigned int x)  // khoang 10.8us
58  {
59    unsigned char i;
60     while(x--)
61    {
62     for (i = 0; i<13; i++)     1/12Mhz * 13 = 10.8us
63    {}
64    }
65  }
```

2. Hàm IrInit cấu hình ngắt và set IRIN lên mức cao tức là khi có tín hiệu truyền đến nó sẽ là IRIN = 0:

```
7   void IrInit()   // Cho phep ngat xay ra
8   {
9       IT0=1;
0       EX0=1;
1       EA=1;
2
3       IRIN=1;    // thiet lap chan IRIN tich cuc muc cao
4   }
5
```

b. Chức năng kết nối LCD:

Đối với chức năng này chúng ta sẽ không đi sau vào chức năng hiển thị lên LCD. Trong file chương trình chính của chúng ta sẽ có 2 file code C:

- File main.c: chứa chương trình chính và phần nhận dữ liệu
- File lcd.c: chưa các thông tin tác vụ xử lí với lcd.

Đối với file lcd.c trong đó sẽ chứa các hàm xử lí lcd 16x2 cơ bản để hỗ trợ việc hiển thị của chức năng hồng ngoại.

c. Chức năng nhận thông tin:

Đi vào chương trình chính "main.c"

Đầu tiên chúng ta sẽ gặp các dòng lệnh khai báo thư viện, các biến và các chân như sau:

```
#include<reg51.h>
#include"lcd.h"
// su dung giao thuc NEC
sbit IRIN=P3^2;

unsigned char code CDIS1[13]={" Red Control "};
unsigned char code CDIS2[13]={" IR-CODE:--H "}; // 2 bien hien thi
unsigned char IrValue[6]; // mang luu gia tri
unsigned char Time;
void IrInit();  // khai bao ham
void DelayMs(unsigned int );
```

Tiếp theo chương trình sẽ dựa vào hàm ngắt (INT0) mỗi khi nhận được dữ liệu 1 từ remote sẽ diễn ra ngắt.

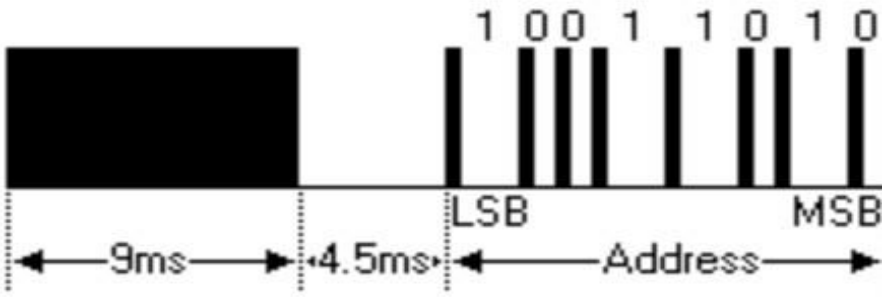
Ở đây mình xin phép được giải thích hàm ngắt trước vì nó chính là hàm nhận thông tin và cũng chính là hàm chính trong chương trình này:

Đây là bước khai báo một số biến cần thiết:

```
void ReadIr() interrupt 0  // ham interrupt
{
    unsigned char j,k;   // bien j,k dung de dem so luong byte va bit
    unsigned int err;     // err dung de kiem soat do tre
    Time=0;               // dung de dem so lan de xac dinh bit 0 hay 1
    DelayMs(70);
```

Tiếp theo chúng ta đến việc giao thức truyền thông ở đây là NEC và ở giao thức này mỗi lần truyền đi nó sẽ truyền đi 4 byte (mỗi byte 8 bit). Ở 2 bit đầu tiên sẽ có 2 bit mở đầu có dạng:

Protocol



Đầu tiên chúng ta có 2 xung mở đầu là 1 dài 9ms và 0 dài 4.5ms

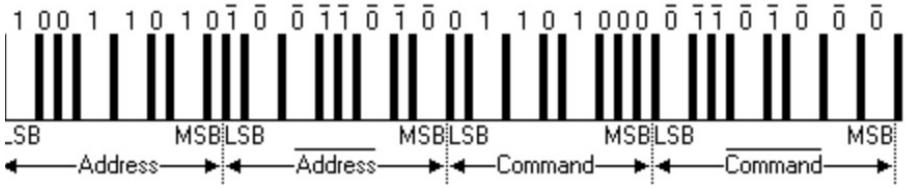
```
if(IRIN==0)     //  nhan tin hieu cao tu remote
{

    err=1000;               //1000*10.8us=10.8ms sap xi 9ms

    while((IRIN==0)&&(err>0)) // giu o trang thai 0 1000 lan
    {
        DelayMs(1);
        err--;
    }
    if(IRIN==1)  // remote truyen den IC tich cuc cao star bit so 2
    {
        err=500;                        // 500*10.8us = 5.4ms sap xi 4.5 ms
        while((IRIN==1)&&(err>0))       // giu o trang thai ngung nhan hoac khi co thay doi
        {
            DelayMs(1);
            err--;
        }
```

Đây là đoạn code dùng để nhận dạng 2 xung đó, ở đây biến IRIN sẽ hoàn toàn ngược với lí thuyết vì mặc định chân ngắt sẽ được set về mức 1.

Tiếp đến chúng ta sẽ đến bước đọc 4 byte bao gồm: 2 byte địa chỉ (0,1) và 2 byte giá trị(2,3):

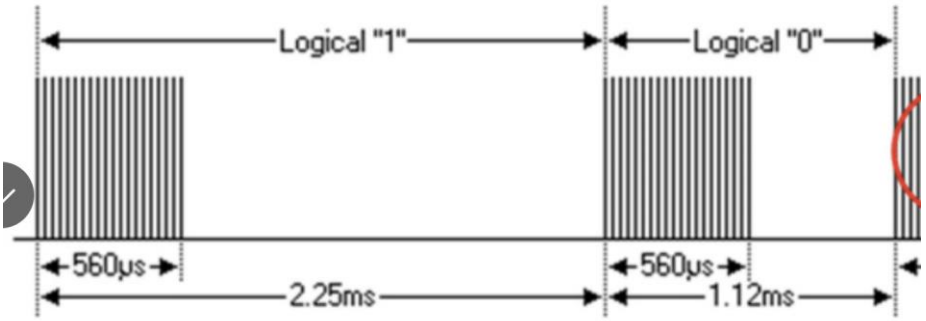



```
    for(k=0;k<4;k++)     // doc 4 byte
    {
        for(j=0;j<8;j++)       // moi byte co 8 bit
        {
```

Bắt đầu đọc lần lượt 8bit của 4byte

```
err=60;      // 60*10.8us = 648 us sap xi 560 ús
while((IRIN==0)&&(err>0))   // nhan bit cao truoc khi den bit thap de nhan dang giua 0 va 0
while (!IRIN)
{
    DelayMs(1);
    err--;
}
err=500;  // 500*10.8 = 5.4ms
while((IRIN==1)&&(err>0))    //bat dau nhan bit nhan dang
{
    DelayMs(1);//10.8us
    Time++;
    err--;
    if(Time>30)   // 30*10.8us = 324us qua thoi gian khong con nhan duoc lenh nao
    {
    EX0=1;   // xay ra ngat --> chuoi gui da khong con
        return;
    }
}
```

Để phân biệt được bit được truyền đến là bit 0 hay 1 thì chúng ta sẽ dựa vào thời gian kéo dài của xung 0 từ remote tức là khi IRIN = 1:



Đây là đại diện cho thời gian kéo dài của bit 0 và 1(cụ thể bit 1 thì IRIN kéo dài khoảng 1.75ms và bit 0 sẽ kéo dài 560us). Tuy nhiên trong lúc khảo sát và nghiêm cứu thì em thấy khi tính Time>30 thì thời gian khá lệch so với lí thuyết có thể là do trong hàm while đó vẫn còn có các hàm khác ảnh hưởng đến thời gian.
Nhưng chúng ta tạm hiểu là khi Time > 30 tức là chuỗi gửi đã hết.
Còn khi Time >= 8 thì đây chính là bit 1

```
if(Time>=8)          // 8*10.8us = 86.4 us khi bit nhan dang du do dai thi set bien dau len 1
{
    IrValue[k]|=0x80;  // dich phai, gia tri dau luon la 1
}
Time=0;     // dat lai time
```

Ngược lại nhỏ hơn 8 sẽ tự tính là bit 0

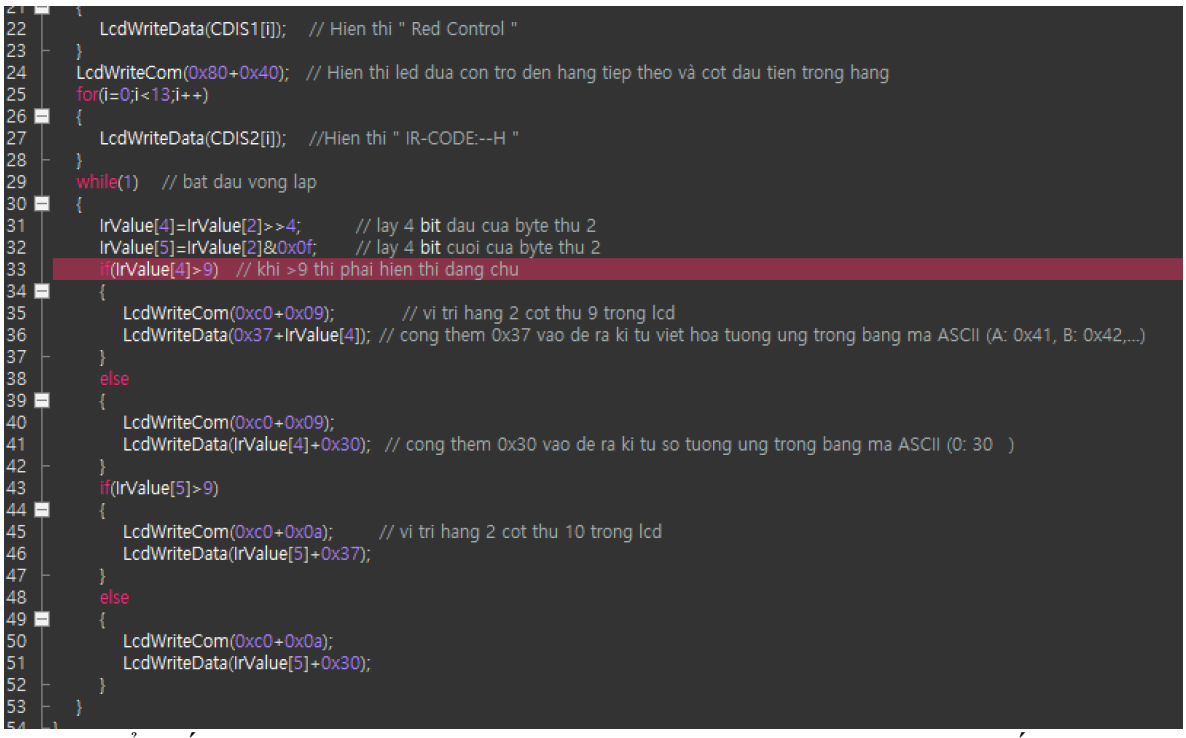
```
IrValue[k]>>=1;    // dich phai, gia tri dau tien luon la 0
```

Đến sau khi nhận đủ 4 byte, 8bit thì chương trình sẽ so sanh 2 bit 2 và 3 để check xem dữ liệu nhận được có đúng không

```
if(IrValue[2]!=~IrValue[3])   // gia tri byte thu 2 khac voi nghich dao cua byte 3, do trong giao thuc NEC byte 2 va byte 3 là 2 byte chua gia tri cua du lieu truyen
```

Nếu giống chúng ta có thể tiến hành xuất thông tin ra LCD.

d. Chức năng xử lí thông tin và xử lí in lên LCD:



```
    {
        LcdWriteData(CDIS1[i]);    // Hien thi " Red Control "
    }
    LcdWriteCom(0x80+0x40);   // Hien thi led dua con tro den hang tiep theo và cot dau tien trong hang
    for(i=0;i<13;i++)
    {
        LcdWriteData(CDIS2[i]);    //Hien thi " IR-CODE:--H "
    }
    while(1)    // bat dau vong lap
    {
        IrValue[4]=IrValue[2]>>4;          // lay 4 bit dau cua byte thu 2
        IrValue[5]=IrValue[2]&0x0f;        // lay 4 bit cuoi cua byte thu 2
        if(IrValue[4]>9)   // khi >9 thi phai hien thi dang chu
        {
            LcdWriteCom(0xc0+0x09);            // vi tri hang 2 cot thu 9 trong lcd
            LcdWriteData(0x37+IrValue[4]); // cong them 0x37 vao de ra ki tu viet hoa tuong ung trong bang ma ASCII (A: 0x41, B: 0x42,...)
        }
        else
        {
            LcdWriteCom(0xc0+0x09);
            LcdWriteData(IrValue[4]+0x30);  // cong them 0x30 vao de ra ki tu so tuong ung trong bang ma ASCII (0: 30   )
        }
        if(IrValue[5]>9)
        {
            LcdWriteCom(0xc0+0x0a);        // vi tri hang 2 cot thu 10 trong lcd
            LcdWriteData(IrValue[5]+0x37);
        }
        else
        {
            LcdWriteCom(0xc0+0x0a);
            LcdWriteData(IrValue[5]+0x30);
        }
    }
```

Để xuất ra LCD khi ta nhận được từ remote là mã BCD của một số ở byte 2 và 3. Thì chúng ta sẽ chuyển nó sang mã hex bằng cách dựa vào mã ASCII. Được giải thích cụ thể trên code trên.