

Tìm hiểu IC 89C2051 và mạch nạp chương trình cho IC 89C2051

I: Khái quát về IC 89C2051:



1. Giới thiệu chung:

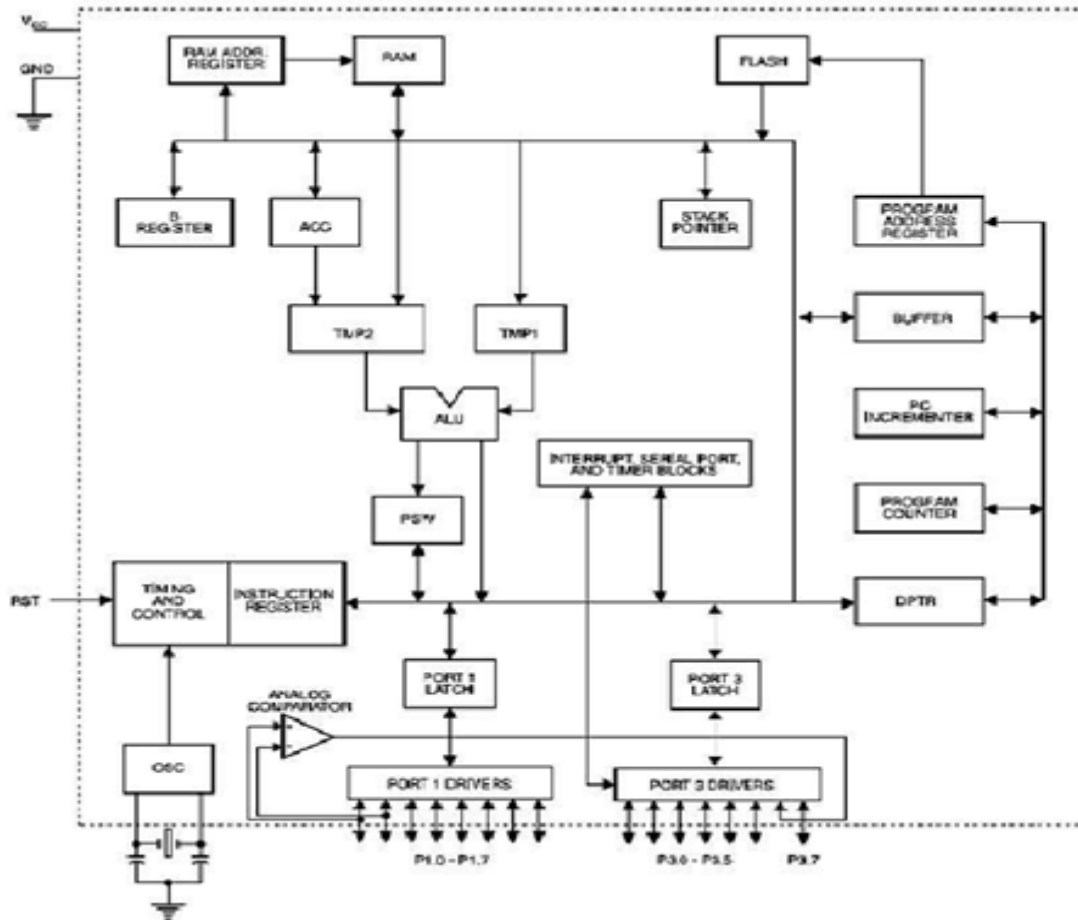
Nếu bạn không muốn dùng con chip 89C2051-40 chân vừa to lại vừa đắt tiền thì bạn có thể dùng con chip 89C2051-20 chân vừa nhỏ gọn vừa tiết kiệm tiền bạc mà vẫn đầy đủ các tính năng như chip 89C51. Chip 89C2051 rất nhỏ gọn nên nó được sử dụng rất nhiều trong các ứng dụng nhỏ. Nếu bạn muốn vừa học VI SỬ LÝ đồng thời cũng muốn khám phá nó qua các ứng dụng cụ thể, qua các dự án thực tế để phát triển 89C2051, 89C4051... với ngôn ngữ lập trình Assembly thì mạch nạp 89C2051 chính là câu trả lời.

2. Một số đặc tính:

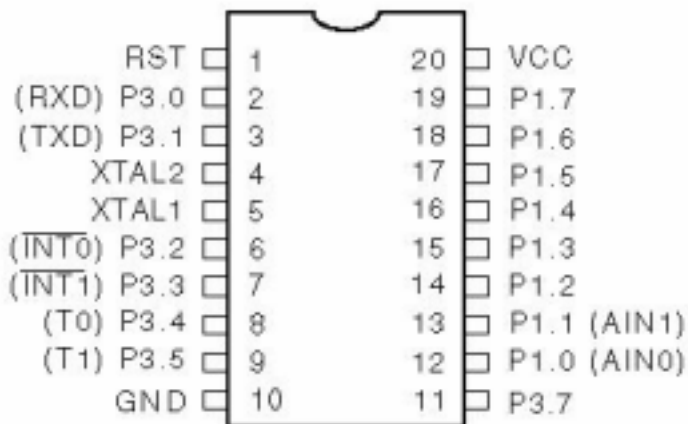
- Đây là một vi điều khiển của hãng atmel, đầy đủ các tính năng như chip 89C51.
- Chip này chỉ có 20 chân. 15 đường xuất nhập
- Điện áp làm việc : 2,7 V → 6V. (Thường dùng ở mức 5V).
- Tần số làm việc: Tần số dao động thạch anh từ 0 tới 24Mhz.
- ROM : 2Kbyte Flash ROM.
- RAM: 128 bytes.
- Hai bộ định thì 16-bit.
- Lập trình tuần tự bằng kênh UART.
- Có 6 nguồn ngắt.
- Có 2 mức khóa bộ nhớ chương trình.
- Có cổng nối tiếp.
- Hai bộ so sánh Analog tích hợp sẵn trên chip.
- Trực tiếp tiếp điều khiển LED ngõ ra.

II. Cấu hình:

1. Sơ đồ khối :



2. Sơ đồ chân :



- PORT 1: Từ chân 12 → 19: Xuất nhập dữ liệu, từ P1.2 → P1.7 được dùng để kéo lên bên trong. P1.0 và P1.1 tương ứng tích cực mức logic cao và thấp cho hai đầu vào AIN0 và AIN1 tương ứng của bộ so sánh chính xác trên chip.

Port 1, bộ khuếch đại đệm đầu ra có thể hạ xuống 20mA và có thể điều khiển LED hiển thị trực tiếp. Chỉ cần 1s để chuyển những chân của Port 1 sử dụng như những đầu vào. Khi chân P1.2 → P1.7 được sử dụng như những đầu vào, chúng sẽ là những nguồn dòng I_{OL} vì được kéo lên bên trong.

Port 1 cũng nhận được mã dữ liệu từ chương trình FLASH và thực hiện.

- PORT 3: Chân số 2, 3, 6, 7, 8, 9, 11, những chân này đã có điện trở kéo lên. P3.6 được nối cố định giữa đường xuất nhập trên bộ so sánh của chip và không thể truy cập. Chỉ cần 1s để chuyển những chân của Port 3 lên mức cao bởi sự kéo lên bên trong và có thể sử dụng như những đầu vào, chúng sẽ là những nguồn dòng I_{OL} vì được kéo lên bên trong. Port 3 cũng phục vụ cho các chức năng của nhiều tính năng đặc biệt của 89C2051 như sau:

| Port Pin | Chức năng thay thế |
|----------|---------------------------------------|
| P3.0 | RXD (chân phát dữ liệu port nối tiếp) |
| P3.1 | TXD (chân nhận dữ liệu port nối tiếp) |
| P3.2 | INT0 (ngắt ngoài 0) |
| P3.3 | INT1 (ngắt ngoài 1) |
| P3.4 | T0 (timer 0 ngõ vào bên ngoài) |
| P3.5 | T1 (timer 1 ngõ vào bên ngoài) |

Port 3 cũng nhận được tín hiệu điều khiển từ Flash và thực hiện.

- Vcc : Chân số 20: điện áp vào khoảng 2,7V → 6V(thường dùng ở mức 5V)

- GND : Chân số 10: chân nối mass.

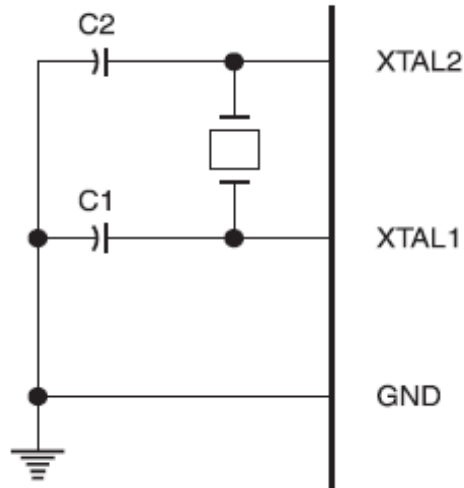
- RST : Xác lập lại trạng thái ban đầu . RST=0: Chip hoạt động bình thường.

RST=1: Chip được thiết lập lại trạng thái ban đầu.

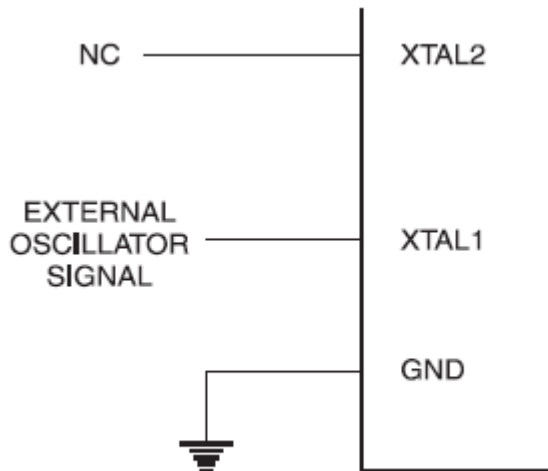
- XTAL1: Ngõ vào mạch tạo xung clock trong chip và ngõ vào bộ khuếch đại đảo chiều.

- XTAL2: Ngõ ra từ bộ khuếch đại đảo chiều.

XYAL1, XTAL2 là ngõ vào và ngõ ra tương ứng của bộ khuếch đại đảo chiều, nó có thể định hình và được sử dụng như một bộ giao động trên chip (hình 1). Tinh thể thạch anh hay cộng hưởng gốm được sử dụng. Hoặc là nhân xung từ bên ngoài(hình 2)



Hình 1: Bộ giao động kết nối.
***NOTE: thạch anh: C1, C2 = 30pF ±10pF**
Cộng hưởng gốm: C1, C2 = 40pF ± 10pF



Hình 2: Nhận xung clock.

3. Thanh ghi có chức năng đặc biệt :

Bên trong sơ đồ của chip có một vùng nhớ đặc biệt được gọi là thanh ghi có chức năng đặc biệt.

Các vùng địa chỉ của thanh ghi được đưa vào bảng dưới đây.

Lưu ý rằng: không phải tất cả các địa chỉ được sử dụng, và các địa chỉ trống có thể không được thực hiện trên chip. Địa chỉ đọc sẽ truy xuất trở về dữ liệu ngẫu nhiên, và địa chỉ ghi sẽ truy xuất về chế độ không có hiệu lực xác định.

Bảng AT89C2051 SFR và thiết lập giá trị:

| | | | | | | | | | | |
|------|------------------|------------------|-----------------|-----------------|-----------------|-----------------|--|--|------------------|------|
| 0F8H | | | | | | | | | | 0FFH |
| 0F0H | B 00000000 | | | | | | | | | 0F7H |
| 0E8H | | | | | | | | | | 0EFH |
| 0E0H | ACC 00000000 | | | | | | | | | 0E7H |
| 0D8H | | | | | | | | | | 0DFH |
| 0D0H | PSW 00000000 | | | | | | | | | 0D7H |
| 0C8H | | | | | | | | | | 0CFH |
| 0C0H | | | | | | | | | | 0C7H |
| 0B8H | IP XXX00000 | | | | | | | | | 0BFH |
| 0B0H | P3 11111111 | | | | | | | | | 0B7H |
| 0A8H | IE 0XX00000 | | | | | | | | | 0AFH |
| 0A0H | | | | | | | | | | 0A7H |
| 98H | SCON 00000000 | SBUF XXXXXXXX | | | | | | | | 9FH |
| 90H | P1 11111111 | | | | | | | | | 97H |
| 88H | TCON 00000000 | TMOD 00000000 | TL0 00000000 | TL1 00000000 | TH0 00000000 | TH1 00000000 | | | | 8FH |
| 80H | | SP 00001111 | DPL 00000000 | DPH 00000000 | | | | | PCON 0XXX0000 | 87H |

4. Bộ nhớ chương trình khóa bit:

Trên chip có hai bộ khóa bit có thể hoạt động không cần lập trình (U), hoặc có thể lập trình (P) để bổ sung thêm nhiều tính năng được liệt kê trong bảng dưới đây.

| Chương trình khóa bit. | | | Loại bảo vệ |
|------------------------|-----|-----|--|
| | LB1 | LB2 | |
| 1 | U | U | Không cần lập trình |
| 2 | P | U | Tính năng cho phép lập trình của Flash bị vô hiệu hóa. |
| 3 | P | P | Tương tự như ở chế độ 2, cũng bị vô hiệu hóa. |

Lưu ý: Những mẫu bit Khóa chỉ được xóa bỏ hoàn toàn bởi Chip Xóa.

5. Chế độ nghỉ :

Ở chế độ nghỉ, CPU được đặt ở chế độ ngủ trong khi tất cả bộ phận ngoại vi vẫn hoạt động. Chế độ này được gọi ra bởi phần mềm. Nội dung của các thanh ghi trong RAM và tất cả các giá trị trong thanh ghi đặc biệt cũng sẽ không đổi ở chế độ này. Chế độ nghỉ có thể bị dừng lại bất kì khi nào có sự kích hoạt hay thay đổi nào đó, hoặc được reset bằng phần cứng.

Các P1.0 và P1.1 nên được thiết lập ở mức "L" nếu bên ngoài-up không được sử dụng, hoặc thiết lập ở mức "H" nếu bên ngoài pull-up được sử dụng.

Cần lưu ý rằng khi “nghỉ” là kết thúc bằng một phần cứng. Tài liệu thực hiện chương trình từ đầu nó lại tắt, lên tới hai chu kỳ máy trước khi các nguyên tắc điều khiển bên trong thiết lập lại. Trên chip phần cứng quyết định quyền truy cập vào bộ nhớ trong RAM trong trường hợp này, nhưng truy cập vào các port không thể quyết định được. Để loại trừ khả năng này xảy ra một cách bất ngờ viết cho một port khi chế độ nghỉ được lặp lại, ta không nên viết tới một Port hay bộ nhớ ngoài

6. Chế độ power-down :

Ở chế độ **power-down**, bộ dao động ngừng, và chương trình sẽ gọi **power-down** và lệnh cuối cùng được thực hiện. Trên chip nội dung RAM và tất cả các giá trị trong thanh ghi đặc biệt cũng sẽ không đổi ở chế độ này cho đến khi chế độ này kết thúc. Chế độ **power-down** chỉ thoát ra khi reset lại phần cứng. Thiết lập lại giá trị các SFR (thanh ghi có chức năng đặc biệt) nhưng trên RAM vẫn giữ nguyên.

Chú ý: Không nên reset lại trước khi VCC được phục hồi lại hoạt động bình thường và phải được giữ mức tích cực đủ dài, để cho phép bộ giao động khởi động lại và làm việc ổn định.

Lưu ý: Ở cả hai chế độ nghỉ và chế độ power-down, P1.0 và P1.1 nên set ở mức "0" nếu không sử dụng điện trở bên ngoài để kéo lên, hoặc set ở mức "1" nếu sử dụng điện trở bên ngoài để kéo lên.

7. Lập trình Flash :

Chip 89C2051 là một loại vi điều khiển với 2K bytes bộ nhớ PEROM có thể xóa hoàn toàn (ví dụ, nội dung = FFH) và có thể lập trình lại. Các mã lập trình bộ nhớ là một mảng byte tại một thời điểm. Sau khi các mảng đã được lập trình, để đảm bảo bất kỳ chương trình nào không trông byte, toàn bộ mảng nhớ cần phải được xóa hoàn toàn bằng điện.

a) Địa chỉ bộ đếm bên trong: Vi điều khiển 89C2051 có một địa chỉ truy cập (bên trong PEROM) địa chỉ đếm luôn luôn đặt ở giá trị 000H trên mức cao của RST và áp dụng mức tích cực của xung dương từ chân XTAL1.

b) Thuật toán: Để lập trình cho chip 89C2051, sau đây là các chuỗi được khuyến cáo nên sử dụng:

1: Chuỗi Power-up :

Áp dụng nguồn điện giữa chân VCC và GND

Đặt RST và XTAL1 để GND

2: Đặt chân RST lên mức cao (mức 1)

Đặt chân P3.2 lên mức cao (mức 1)

3: Áp dụng kết hợp giữ 2 mức logic “H” hoặc “L” ; (“1” hoặc “0”)

tới cho các chân P3.3, P3.4, P3.5, P3.7 để lựa chọn một trong những chương trình hoạt động hiển thị trong PEROM bảng chế độ lập trình dưới đây.

4: Áp dụng cho dữ liệu mã byte từ vị trí 000H đến P1.0 đến P1.7.

5: Cho RST lên 12V để kích hoạt chương trình.

6: Xung từ chân P3.2 tới chương trình một byte ở trong PEROM hoặc bit khóa. Các byte-ghi là chu kỳ tự hẹn giờ và thường mất trong 1,2 ms.

7: Để kiểm tra dữ liệu được lập trình, thấp hơn RST từ 12V, ta để mức logic "1" và set chân P3.3 đến P3.7 giữ ở mức thích hợp. Dữ liệu ra có thể đọc ở Port 1.

8: Để lập trình một byte ở vị trí kế tiếp, xung kích từ chân XTAL1 được kích một lần để nâng cao số bộ định địa chỉ bên trong. Dữ liệu mới được đưa vào Port 1.

9: Lặp lại các bước 6 thông qua bước 8, thay đổi dữ liệu và nâng cao địa chỉ truy cập cho toàn bộ 2K bytes mảng hoặc cho đến khi kết thúc đối của tập tin là được.

10: Chuỗi Power-off:

XTAL1 và RST set ở mức "L".

Kiểm tra dữ liệu: chip AT89C2051 sẽ kiểm tra tuần tự dữ liệu để và cho biết thời điểm kết thúc của một chu kỳ viết. Trong thời gian một chu kỳ máy, nó sẽ cố đọc tới byte được ghi cuối cùng và sẽ bổ sung các byte dữ liệu trên P1.7. Sau khi chạy xong 1 chu kỳ máy, thấy dữ liệu hợp lệ ở tất cả các port, nó sẽ bắt đầu chạy chu kỳ kế tiếp. Việc kiểm tra có thể bắt đầu bất cứ lúc nào khi chu kỳ kế tiếp được tiến hành

READY / BUSY (sẵn sàng/bận): Byte tiến trình của chương trình cũng có thể được theo dõi bởi tín hiệu đầu ra READY/BUSY. Chân P3.1 ở mức thấp sau khi chân P3.2 ở mức cao trong thời gian chương trình thực hiện để báo BUSY (bận). chân P3.1 sẽ trở lại mức cao khi chương trình thực hiện để báo READY (sẵn sàng).

Chương trình kiểm tra : Nếu bit khóa LB1 và LB2 chưa được lập trình mã dữ liệu thì có thể đọc lại dữ liệu thông qua các đường dây để kiểm tra:

1: Thiết lập lại địa chỉ truy cập bên trong là 000H và chân RST từ mức L lên mức H.

2: Áp dụng việc kiểm tra các tín hiệu điều khiển cho phép đọc mã dữ liệu và đọc các dữ liệu xuất ra từ Port 1.

3: Xung kích từ chân XTAL1 được kích 1 lần để nâng cao số bộ định địa chỉ bên trong.

4: Đọc tiếp dữ liệu mã byte tiếp theo tại ngõ ra Port 1.

5: Lặp lại các bước 3 và 4 cho đến khi đọc hết toàn bộ mảng.

Bit khóa không thể kiểm tra trực tiếp, mã xác nhận của bit khóa xác định được bằng cách quan sát những tính năng của chúng.

Chip xóa : toàn bộ mảng PEROM (2KB) và 2 bộ Look Bit cần được xóa hoàn toàn bằng tín hiệu điện bằng cách kết hợp chính xác tín hiệu điều khiển và bằng cách giữ tín hiệu chân P3.1 ở mức thấp trong 10ms. Mã mảng phải viết tất cả ở mức H trong lúc chip xóa

làm việc, và phải thực hiện trước khi bất kỳ byte trống nào trong bộ nhớ được lập trình lại.

Đọc kí hiệu byte: Kí hiệu byte được đọc bình thường và kiểm tra địa chỉ 000H, 001H, và 002H, ngoại trừ P3.5 và P3.7 phải được đặt ở mức logic thấp.

Các kết quả như sau:

(000H) = 1EH chỉ sản xuất bởi Atmel

(001H) = 21H cho biết 89C2051

8. Giao diện lập trình: Mọi mã byte trong mảng Flash được ghi và toàn bộ mảng có thể xóa bỏ bằng cách sử dụng kết hợp thích hợp của các tín hiệu điều khiển. Ghi chu kỳ hoạt động là tự hẹn giờ và sau mỗi lần triển khai sẽ tự động điều chỉnh phù hợp thời gian để hoàn thành.

9. Chế độ lập trình flash :

| Chế độ | | RTS/VPP | P3.2/programe | P3.3 | P3.4 | P3.5 | P3.7 |
|------------------|-------|---------|---------------|------|------|------|------|
| Viết mã data | | 12V | | 0 | 1 | 1 | 1 |
| Đọc mã data | | 1 | H | 0 | 0 | 1 | 1 |
| Write look | Bit 1 | 12V | | 1 | 1 | 1 | 1 |
| | Bit 2 | 12V | | 1 | 1 | 0 | 0 |
| chip xóa | | | 12V | (2) | 1 | 0 | 00 |
| Đọc kí hiệu byte | | | H | 0 | 0 | 0 | 0 |

Lưu ý rằng: - Địa chỉ Ram nội PEROM được thiết lập với giá trị 000H trên mức cạnh tích cực của RST, và được nâng cao do mức tích cực của xung tại XTAL1.

- Chip xóa đòi hỏi phải có 10ms xung chương trình.

- P3.1 phải để mức thấp trong thời gian lập trình để cho biết là READY / BUSY.(sẵn sàng/bận)

10. Đặc tính làm việc DC :

$T_A = -40^{\circ}\text{C} \rightarrow 80^{\circ}\text{C}$, $V_{cc} = 2,7\text{V} \rightarrow 6\text{V}$.

| Kí hiệu | Tham số | Điều kiện | Min | Max | Đơn vị |
|----------|---|--|-------------------|-------------------|-----------|
| V_{IL} | Điện áp vào mức thấp | | - 0,5 | $0,2V_{CC} - 0,1$ | V |
| V_{IH} | Điện áp vào mức cao | (loại trừ XTAL1, RST) | $0,2V_{CC} + 0,9$ | $V_{CC} + 0,5$ | V |
| $VIH1$ | Điện áp vào mức cao | (XTAL1, RST) | $0,7 V_{CC}$ | $V_{CC} + 0,5$ | V |
| V_{OL} | Điện áp ra mức thấp (Port 1, 3) | $I_{OL} = 20 \text{ mA}, V_{CC} = 5V$ $I_{OL} = 10 \text{ mA}, V_{CC} = 2,7V$ | | 0,5 | V |
| V_{OH} | Điện áp ra mức cao (Port 1, 3) | $I_{OH} = -80 \mu A, V_{CC} = 5V \pm 10\%$ | 2,4 | | V |
| | | $I_{OH} = -30 \mu A$ | $0,7 V_{CC}$ | | V |
| | | $I_{OH} = -12 \mu A$ | $0,9 V_{CC}$ | | V |
| I_{IL} | Dòng vào logic 0 (Ports 1, 3) | $V_{IN} = 0,45V$ | | - 50 | μA |
| I_{TL} | Dòng chuyển tiếp từ mức 0 -> 1 (Ports 1, 3) | $V_{IN} = 2V, V_{CC} = 5V$ $\pm 10\%$ | | -750 | μA |
| I_{LI} | Dòng vào rò rỉ (Port P1.0, P1.1) | $0 < V_{in} < V_{CC}$ | | ± 10 | μA |
| VOS | so sánh độ lệch áp vào | $V_{CC} = 5V$ | | 20 | mV |
| RRST | Reset điện trở kéo xuống | | 50 | 300 | $K\Omega$ |
| CIO | Chân tụ | $f = 1 \text{ MHz}, T_A = 25^\circ C.$ | | 10 | pF |
| I_{CC} | Dòng cung cấp hiện thời | Chế độ làm việc, 12MHz, $V_{CC} = 6V/3V$ | | 15/5,5 | mA |
| | | Chế độ nghỉ, 12MHz, $V_{CC} = 6V/3V, P1.0 \& P1.1 = 0$ | | 5/1 | mA |
| | Chế độ power-down | $V_{CC} = 6V, P1.0 \& P1.1 = 0$ | | 100 | μA |
| | | $V_{CC} = 3V, P1.0 \& P1.1 = 0$ | | 20 | μA |

Ghi chú:

- Điều kiện để trạng thái ổn định là I_{OL} phải ở giới hạn ngoài những hạn chế sau :

$$I_{OL} \text{ max} = 20\text{mA.}$$

Tổng dòng cực đại của I_{OL} và các chân ngõ ra là 80mA.

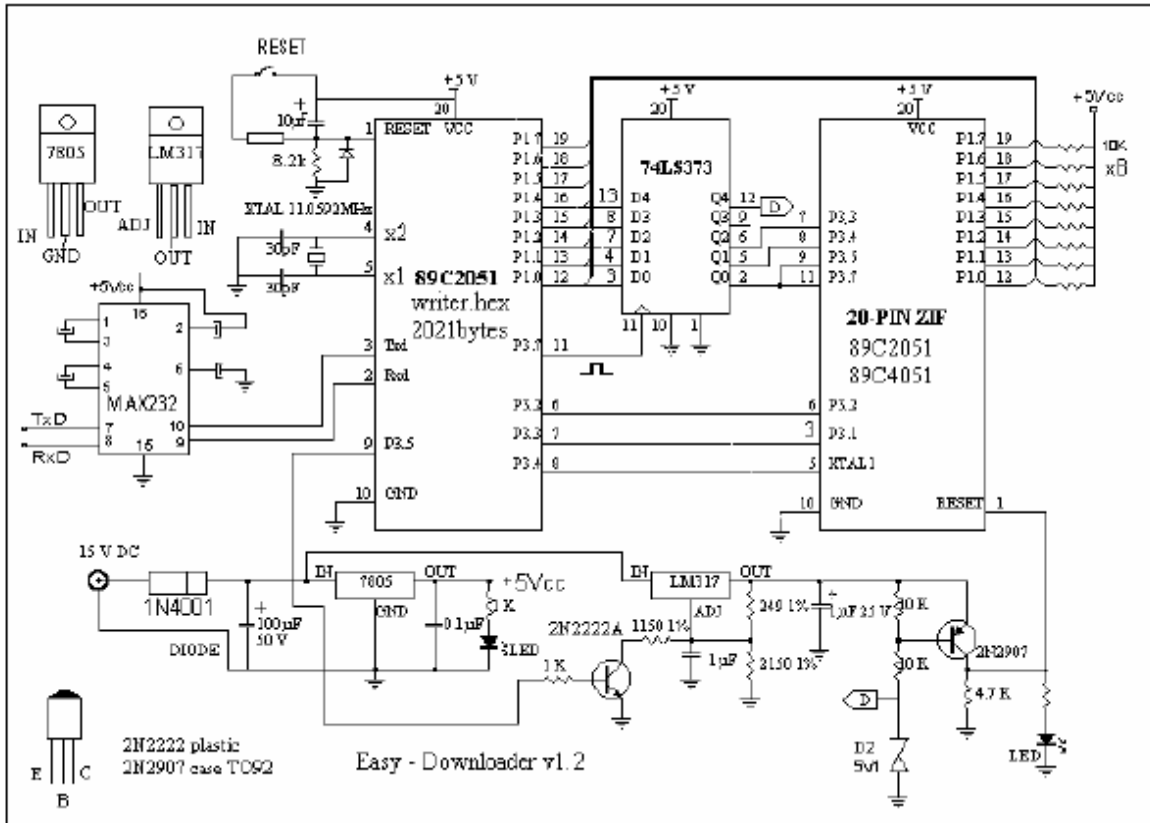
Nếu I_{OL} vượt quá điều kiện cho phép, V_{OL} có thể vượt qua các tiêu chuẩn kỹ thuật liên quan của chip. Các chân chip không được đảm bảo khi dòng lớn hơn điều kiện cho phép.

- Vcc nhỏ nhất của chế độ power-down là 2V.

Mạch nạp IC 89C2051

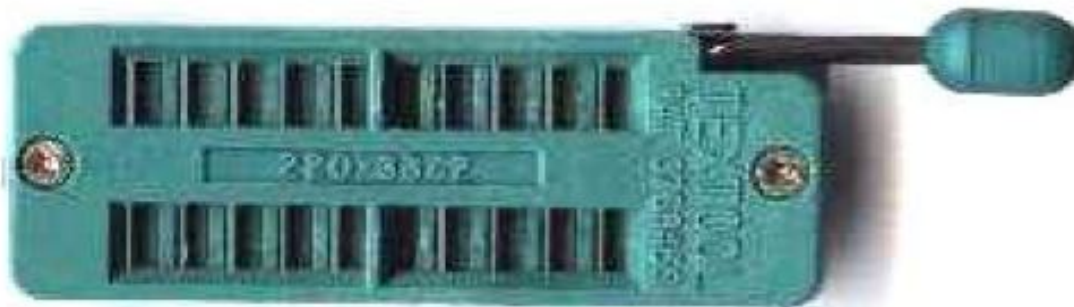
A) Phần cứng:

1: Sơ đồ của mạch nạp IC 89C2051.



Trên sơ đồ bạn hãy lưu ý hai điểm sau:

- Kí hiệu IC ghi 89C2051 Writer.hex 2021 byte gọi là chip chính(Chip master).
- Kí hiệu IC ghi 20-PIN ZIF Socket 89C2051/4051: Đây là cái Socket hay là cái chân để cắm IC, gọi là con chip phụ(Chip Slave).
- Đây là hình dáng thật của Socket 20 chân:



Hãy nhớ rằng: trước khi nạp cho IC, bạn phải nạp file Writer.hex cho con ChipMaster đã:

Có hai file Writer.hex sau: Các bạn có thể tải về theo địa chỉ sau:

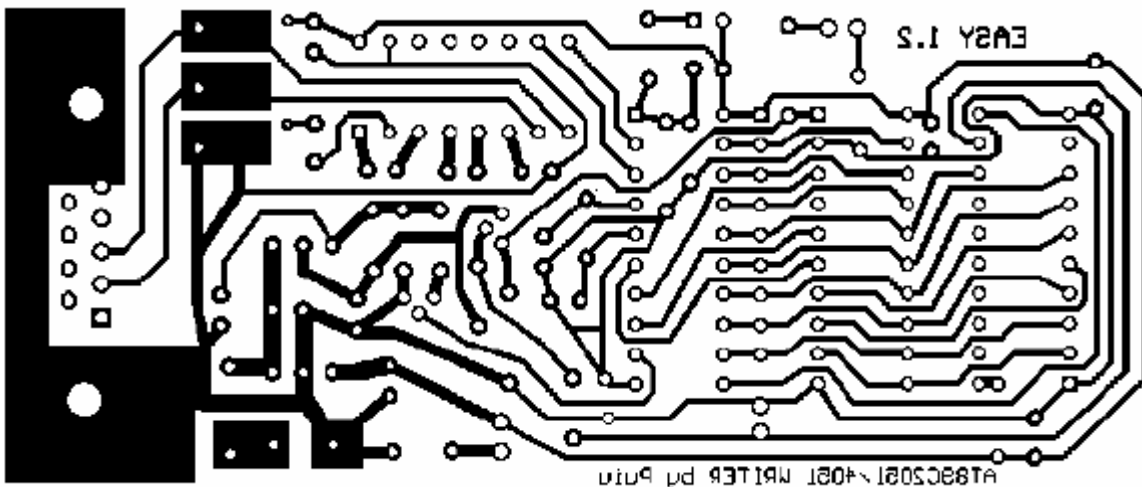
Write1.hex: <http://www.mediafire.com/?mufemhmxwku>

Write.hex: <http://www.mediafire.com/?qmdetynunj5>

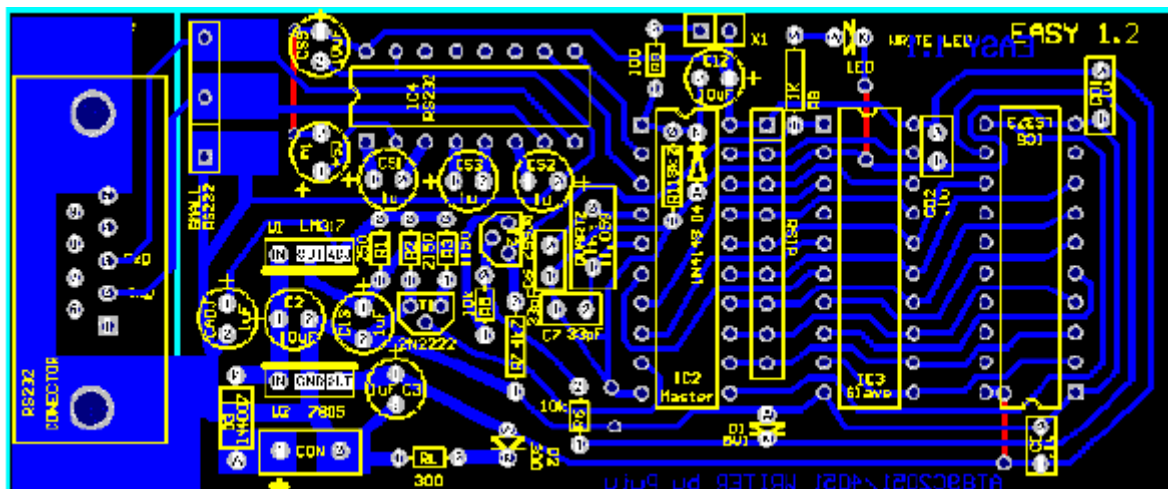
2: Bo mạch in của mạch nạp IC 89C2051:

Dưới đây tôi xin giới thiệu bo mạch qua cổng COM. Các bạn cũng có thể tự thiết kế cho mình những bo mạch khác.

- Hình ảnh hướng dẫn lắp ráp linh kiện lên bo mạch:



- Bo mạch in của mạch nạp:



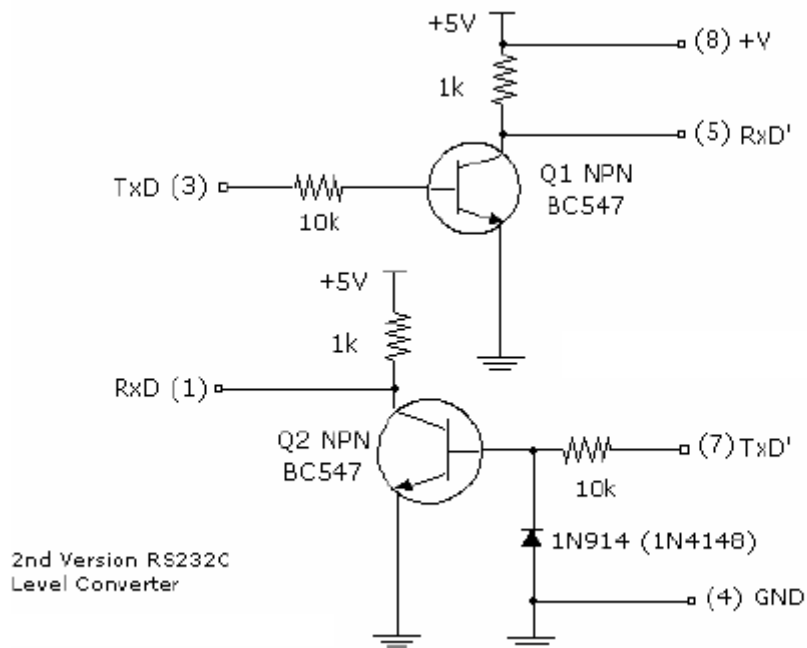
- Hình ảnh của mạch nạp hoàn chỉnh:

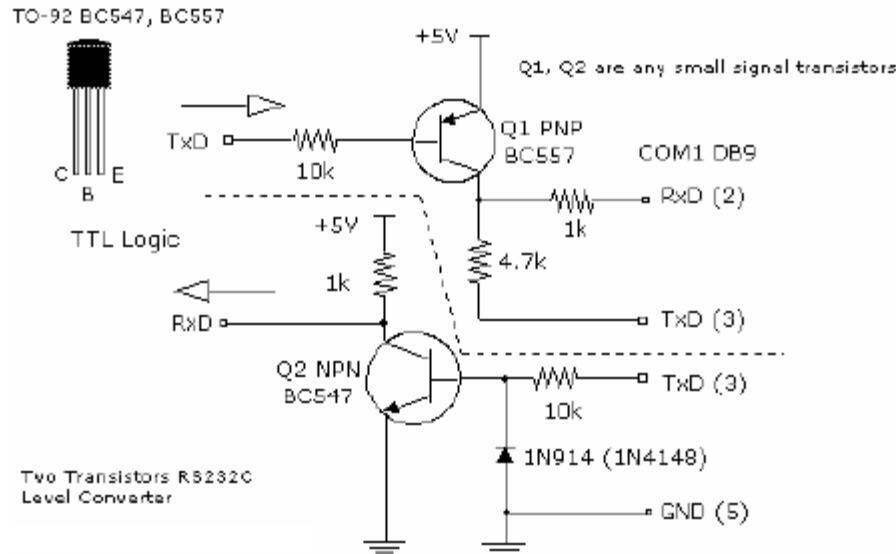


Hãy nhớ rằng: MAX 232 16 chân có thể được thay thế bằng 2 con transistor, hoặc bằng con chip DS2578 chân hết sức đơn giản như sau:

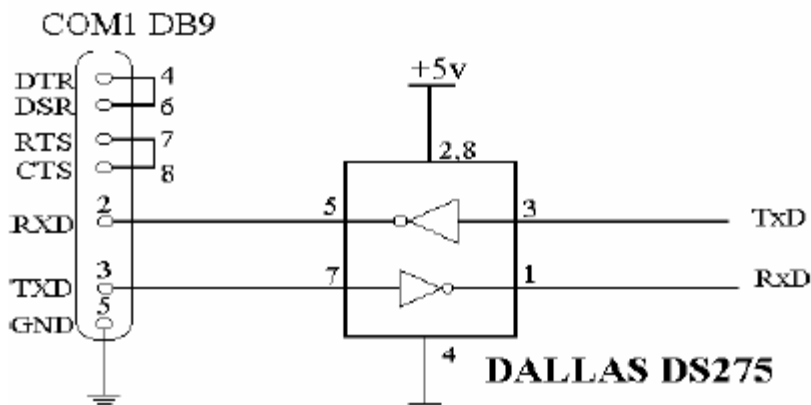
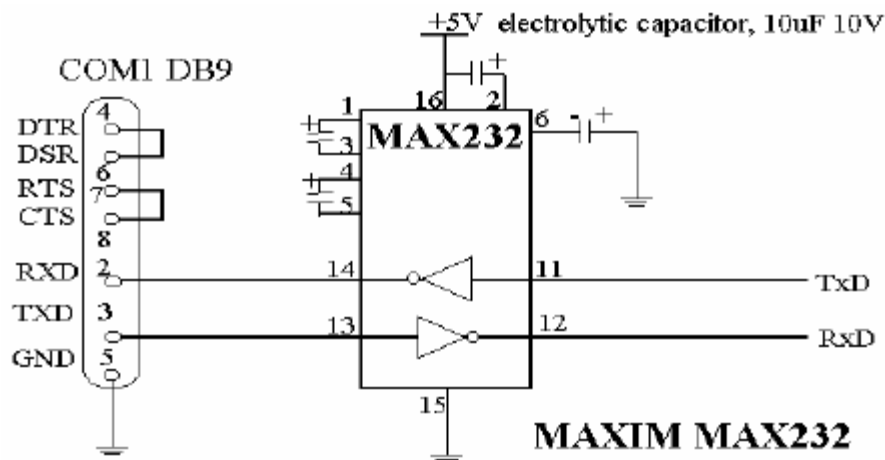
Sơ đồ thay thế như sau:

-Cách dùng hai transistor:





- Cách dùng chip DS257:



B) Phần mềm

1: Các phần mềm biên dịch .ASM sang .HEX

a) Giới thiệu: Như chúng ta đã biết, muốn nạp được nội dung chương trình mà chúng ta đã lập trình vào trong con vi sử lý để nó hoạt động thì chúng ta phải nạp vào cho nó các file có dạng là file **.HEX** hoặc là file **.BIN**. Nhưng đa số người ta thường sử dụng file **.HEX**. Do vậy, tôi xin giới thiệu với các bạn các phần mềm thường dùng để chuyển file **.ASM** sang file **.HEX**

b) Phần mềm ASM51:

- Các file cần có trong bộ ASM51:

ASM51.EXE The Cross Assembler program itself
MOD152 Source file for the \$MOD152 control
MOD154 Source file for the \$MOD154 control
MOD252 Source file for the \$MOD252 control
MOD44 Source file for the \$MOD44 control
MOD451 Source file for the \$MOD451 control
MOD452 Source file for the \$MOD452 control
MOD51 Source file for the \$MOD51 control
MOD512 Source file for the \$MOD512 control
MOD515 Source file for the \$MOD515 control
MOD517 Source file for the \$MOD517 control
MOD52 Source file for the \$MOD52 control
MOD521 Source file for the \$MOD521 control
MOD552 Source file for the \$MOD552 control
MOD652 Source file for the \$MOD652 control
MOD751 Source file for the \$MOD751 control
MOD752 Source file for the \$MOD752 control
MOD851 Source file for the \$MOD851 control

- **Cách sử dụng:** Sau khi lập trình xong (bằng *Notepad* hay *NC-Edit*) bạn hãy lưu nó lại với tên là *****.ASM** (******* là tên mà bạn đặt). Giả sử là: vidu.asm và file này bạn lưu trong ổ C. Đồng thời tôi cũng giả sử rằng bạn cũng để bộ ASM51 này trong ổ C, thì khi đó chúng ta có như sau:

C:\ vidu.asm

ASM51.EXE The Cross Assembler program itself
MOD152 Source file for the \$MOD152 control
MOD154 Source file for the \$MOD154 control
MOD252 Source file for the \$MOD252 control
MOD44 Source file for the \$MOD44 control
MOD451 Source file for the \$MOD451 control
MOD452 Source file for the \$MOD452 control
MOD51 Source file for the \$MOD51 control
MOD512 Source file for the \$MOD512 control
MOD515 Source file for the \$MOD515 control

MOD517 Source file for the \$MOD517 control
MOD52 Source file for the \$MOD52 control
MOD521 Source file for the \$MOD521 control
MOD552 Source file for the \$MOD552 control
MOD652 Source file for the \$MOD652 control
MOD751 Source file for the \$MOD751 control
MOD752 Source file for the \$MOD752 control
MOD851 Source file for the \$MOD851 control

Sau đó bạn hãy mở cửa sổ **MS-DOS** ra và gõ lệnh như sau thì bạn đã có một file có tên là : **vidu.HEX**: Dòng lệnh như sau:

C:\asm51 vidu.asm hoặc C:\asm51 vidu

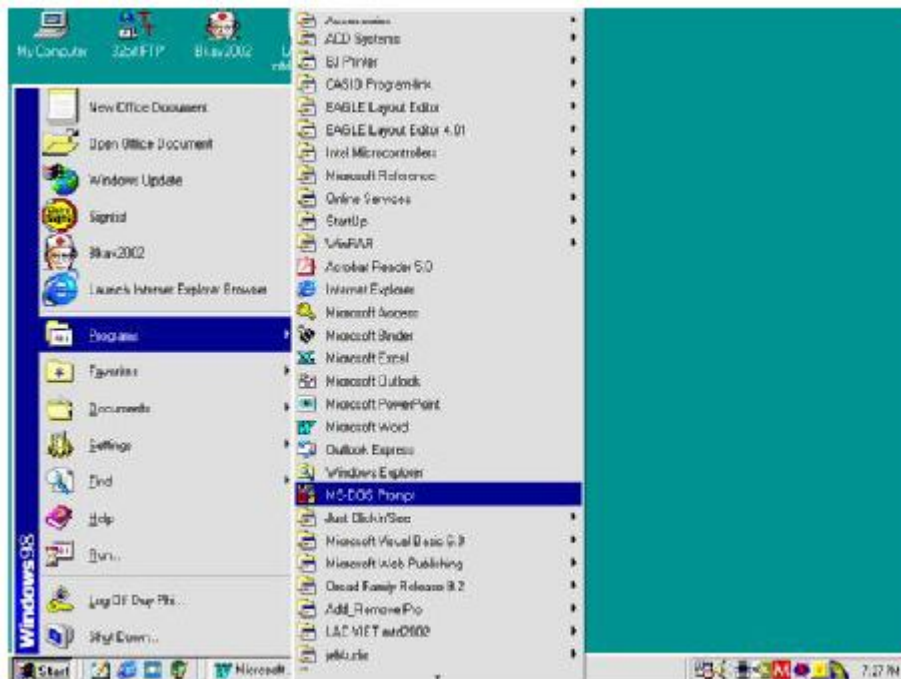
***** bạn có thể vào trang www.visuly.cjp.net để download phần mềm này về máy của mình*****

c) Phần mềm TASM:

- Nội dung các file có trong bộ ATSM:

Bộ TASM chứa trong một file có tên là TASMINST có nghĩa là TASMISTAL (tệp tin để cài đặt), bạn click chuột lên tệp tin này thì nó sẽ tự động giải nén và cài đặt luôn, dung lượng của tệp tin này là 126Kb, các bạn có thể tải miễn phí trên trang: www.visuly.cjp.net

- **Cách sử dụng:** Bạn hãy thoát khỏi môi trường Window ra môi trường DOS bằng cách từ môi trường Window bạn click chuột vào Start menu chọn mục Program chọn MS-DOS prompt.

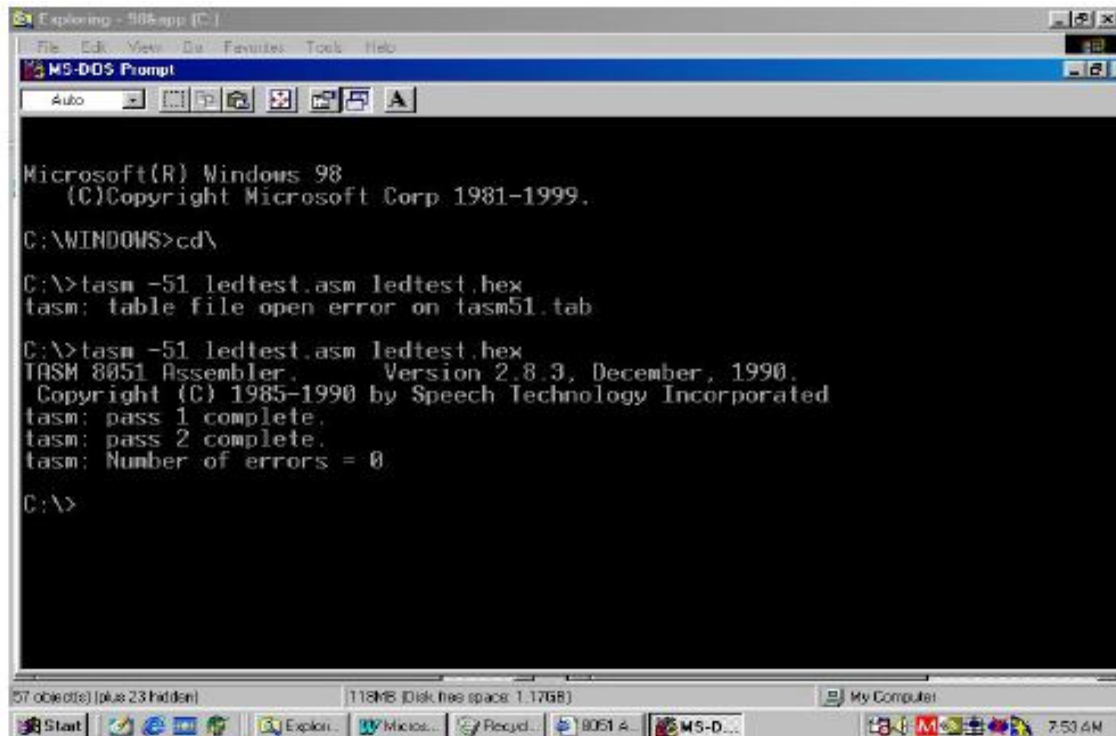


Sau đó bạn hãy thay đổi lại đường dẫn, cùng thư mục để đến nơi chứa file TASM để bắt đầu biên dịch bằng cách sử dụng dòng lệnh sau:

```
Tasm-51 *asm *hex
```

Tôi giả sử bạn để file phần mềm TASM và file vidu.asm tại ổ C, thì khi đó bạn hãy gõ đúng dòng lệnh sau:

```
C:\ tasm-51 vidu.asm vidu.hex
```



```
Microsoft(R) Windows 98
(C)Copyright Microsoft Corp 1981-1999.
C:\WINDOWS>cd\
C:\>tasm -51 ledtest.asm ledtest.hex
tasm: table file open error on tasm51.tab
C:\>tasm -51 ledtest.asm ledtest.hex
TASM 8851 Assembler,      Version 2.8.3, December, 1990.
Copyright (C) 1985-1990 by Speech Technology Incorporated
tasm: pass 1 complete.
tasm: pass 2 complete.
tasm: Number of errors = 0
C:\>
```

Vậy là bạn đã tạo được file vidu.hex rồi đó.

Hãy lưu ý rằng: Bạn cũng có thể tạo ra tệp tin kiểm tra lỗi lập trình trước khi tạo ra tệp tin để nạp cho con vi xử lý bằng cách là bạn tạo ra file có đuôi là **.Lst**. Tệp tin này sẽ kiểm tra lỗi cú pháp lệnh trong chương trình của bạn. Nó sẽ chỉ ra những lỗi sai cho bạn để kịp thời sửa chữa khi đó bạn hãy gõ câu lệnh sau:

```
tasm-51 *asm *obj
```

Kết quả sẽ được hai tệp tin: ***.Obj** và ***.Lst**

Cũng với giả sử như trên ta sẽ gõ như sau:

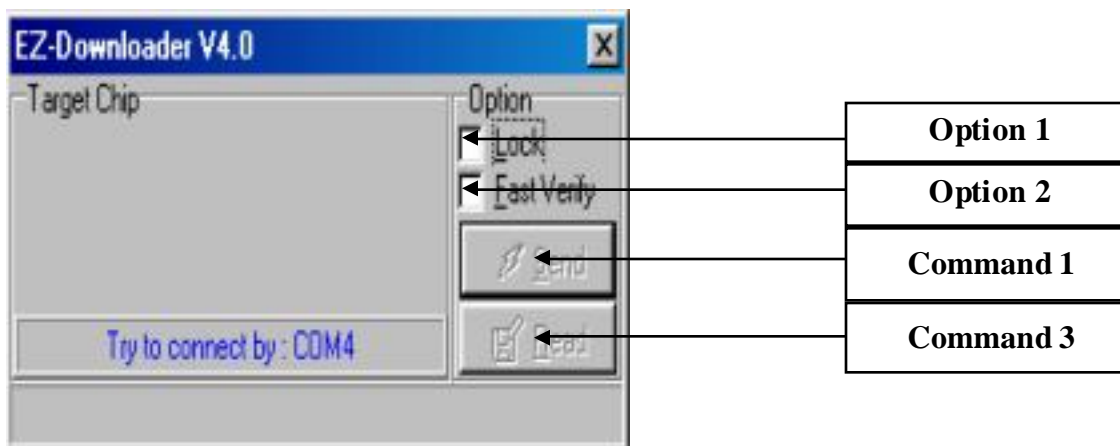
```
C:\ tasm-51 vidu.asm vidu.obj
```

Sau đó bạn sẽ được hai tệp tin: vidu.Obj và vidu.Lst. Bạn hãy mở file vidu.Lst này bằng Notepad hay Word để xem kết quả kiểm tra.

2: Phần mềm nạp cho IC89C2051.

a) Phần mềm EZ4.0 (qua cổng COM) : Đây là phần mềm mới nhất và hoàn toàn miễn phí chạy trên nền Windows rất đẹp và dễ dàng sử dụng.

+ **Giao diện của nó như sau:**



+ **Cách sử dụng:**

- **Option 1:** Khi bạn chọn nút này thì con chip của bạn nó sẽ khóa lại, sau này bạn sẽ không thể nào nạp lại được nữa, do đó rất ít ai chọn nút này. Theo tôi thì không chọn là tốt hơn, để có thể nạp nhiều hơn..

- **Option 2:** Khi bạn chọn nút này là việc kiểm tra sau khi nạp xong sẽ xảy ra rất nhanh. Bạn có chọn hay không chẳng ảnh hưởng gì cả.

- **Command 1:** Chức năng của command này là nạp chương trình file.hex vào cho vi xử lý. Bạn hãy nhấn vào command này để chỉ đến nơi chứa file.hex cần nạp để tiến hành quá trình nạp.

*** Lưu ý là khi bạn chọn command này thì điều đầu tiên nó sẽ xóa nội dung cũ trên con vi xử lý, để dọn đường chuẩn bị nạp nội dung mới vào. Công việc này nó sẽ làm tự động hoàn toàn.

-**Command 2:** Chức năng này sẽ đọc ngược nội dung có trong con vi xử lý của bạn ra ngoài dưới dạng file.hex.

*****HÃY NHỚ RẰNG: Phần mềm này dùng chung được cho cả hai mạch nạp 89C51 và 89C2051. Việc nhận dạng sẽ được nó làm tự động hoàn toàn*****

b) Phần mềm WinATP (qua cổng máy in):

Mạch này sử dụng phần WinATProg chạy trên môi trường Windows. Giao diện như sau:



Bạn có thể chọn được loại chip để nạp trong mục Prozessor.
89C1051 - *89C2051* - *89C4051*



Bạn cũng có thể chọn lại ngôn ngữ hiển thị trên giao diện là tiếng anh.

