Hướng dẫn sửa lỗi do set sai Fuse Bit avr (atmega8, atmega328...)

[Graphical user interface, application

Description automatically generated](https://2.bp.blogspot.com/-uEtJsv7h3Zc/XZOAxl0mN8I/AAAAAAAAlxc/U-IurBL3v94asLeeXGVJS8fdDt6mRvOxACK4BGAYYCw/s1600/17-single-default.jpg)  
**Fuse Bit là gì , Làm sao phải SET nó ?**  
Như chúng ta đều biết, fuse bit được set qua các loại mạch nạp cho AVR, chúng có vai trò quan trọng, chỉ thị MCU hoạt động theo một số quy tắc như dùng thạch anh hay dùng xung nội, vị trí của bootloader, có dùng bootloader hay không, etc. Khi ta mua một con ATmega brand-new về thì mặc định của nó sẽ xài xung nội và ai cũng muốn xài con thạch anh mới mua bên cạnh :)) . Newbie cũng không ngoại lệ và họ cũng tập tành set fuse nhưng tôi cá là sau vài lần set thể nào cũng có cú set sai, và lần sau đọc, ghi sẽ bung ra lỗi này:  
  
Fuse bit là các bit cứng nhất với vi điều khiển của bạn , đó là các bit cần thết nhất để một vi điều khiển có thể hoạt động tốt bởi nó quy định phần cứng (tần số thạch anh , có hay không sử dụng chân RESET , Quy định điện áp RESET , và có sử dụng mạch debug Jtag không ) và còn quy định một số modul bên trong chip (Wotdog timer , SPI , và cả bộ nhớ bootloader và có boot vào bootder không ).  
  
Các Bit cần chú ý để Set :  cái hay ở đây là ATMEL quy định với FUSE bit , Bit 0 là có sử dụng , Bit 1 là không sử dụng .  
[Table

Description automatically generated](https://3.bp.blogspot.com/-SY3WMqkMxTs/XZOA1cXQCbI/AAAAAAAAlxk/7QHyTQpcwYMYZ38Pr3KM6OTmXhxIYzPJgCK4BGAYYCw/s1600/set_fuse_bit2.png)  
  
**Ở các bit thấp :**  
  
BODLEVER là bit chọn mức điện áp tham chiếu sụt áp nguồn, khi ta cấu hình kích hoạt BODEN. BODLEVER=0 tức chọn mức điện áp 4.0V, còn nếu BODLEVER=1 là chọn mức điện áp 2.7V (có thể tùy vào dòng chíp bạn cầ tra datasheet ) Khi điện áp trên chip bị sụt quá mức thiết đặt trên thì chip tự RESET không như một số vi điều khiển khác bị treo luôn .  
  
BODEN là bit cho phép mạch phát hiện sụt áp nguồn hoạt động. Sut áp nguồn xảy ra  
  
SUT1:0  :thời gian khởi động chip , ko dùng và mặc định như nào để đó SUT1:0=10.  
  
CKSEL3:0 :  Chọn xung nhịp : mỗi chíp có một cách chọn khác nhau bạn nên tra datasheet để set  
 **Một số xung nhịp hay dùng :**  
[Table

Description automatically generated](https://2.bp.blogspot.com/-cyooFF6wX3o/XZOA6Ygzr6I/AAAAAAAAlxs/Zi87xk_ZENEmLWK2IKRFUlS_qSBNlp3kwCK4BGAYYCw/s1600/set_fuse_bit3.png)  
  
**Ở các Bit cao :**  
  
OCDEN (on chip debug enable ): avr cho phép bạn gỡ rối , debug chương trinh ngay trên chip như trong các con chip ARM thần thánh của STM như STM32F4 . AVR ở đây sẽ được debug bằng giao tiếp Jtag trên một số mạch nạp trên thị trường đã có , để sử dụng tính năng Debug bạn cần set bit OCDEN ở trạng thái 0 .  
  
JTAEGE (JTAG Enable ) : Bit này để bạn cho phép kết nối Jtag (Kết nối bởi 4 chân TCK, TMS, TDO và TDI ) sử dụng để nạp mạch bằng mạch nạp jtag và debug , khi set bit này bằng 0 , các chân IO ứng với các chân này bạn không thể sử dụng để xuất nhập dữ liệu . Nếu bạn set JTAEGE = 1 mà ban Set OCDEN =0 thì bạn vẫn không thể debug chương trình được .  
  
SPIEN (Enable Serial programming and Data Downloading) : cho phép sử dụng modul SPI ở trong chip , nếu không set bit này bằn 0 thì bạn có cấu hình thần thánh SIP bằng code thì vẫn không thể sử dụng được .  
  
CKOPT (Oscillator Options) : Bit quy định tần số thạch anh ngoại lớn nhất  cần tra datasheet của mỗi con chip  các lỗi fuse bit chủ yếu xảy ra với bit này. (nếu sử dụng tần số thạch anh ngoài lớn thì set nó =0 )  
  
EESAVE (EEPROM memory is preserved through chip erase) : cho phép xóa bộ nhớ EEROM không khi xóa để ghi và nạp chip . (bit này = 0 thì không được xóa  )  
  
BOOTSZ 0..1 : hai bit quy định độ rộng cho bộ nhớ Bootloader Tra datasheet để set .  
  
BOOTRST (Select Reset Vector ) Quy định khi bắt đầu chương trình con trỏ chương trình sẽ chạy ở vị trí nào , 0x0000 hay vùng Bootloader (nếu bit này bằng 0 thì khi khởi động hoặc nhấn Reset chương trình sẽ thược thi ở vùng nhớ Bootloader )  
  
**Một số bit gia tăng :**  
  
WDTON (Watchdog timer always on) : cho phép chế độ Watdog  timer hoạt động (Reset khi chíp không làm gì ở một số thời gian được set bằng code )  
  
Bạn có thể kham khảo Fuse bit của một số chip tại : http://www.engbedded.com/fusecalc  
 **Nguyên nhân:**  
Nguyên nhân thì nhiều: do thiếu hiểu biết, do muốn vọc vạch, muốn xem set sai nó ra cái gì, etc (với tôi thì là tổng hợp của những cái trên, cộng thêm cái PROGISP này bit 1-0 nó ngược với tut tôi đọc trên hocavr nên ăn hành là khó tránh).  
  
Lỗi ở đây chủ yếu là bạn chọn sai bộ giao động làm cho con chip không phân biệt được hay lỗi các phần giao động. Bạn có thể dử dụng các mạch nạp có thể tạo dao động và nạp chip để set lại fuse bit . Hoặc có các là bạn vẫn cắm mạch ISP đang sử dụng và dùng bất kì modul giao động nào đó , có thể là một con chip đang sử dụng được cho giao động ở tần số bất kì lớn hơn 1MHZ  và đặt giao động đó vào chân XTAL1 (đúng chân ) không cần tháo thạch anh và nhớ câu 2 dây GND với nhau để đồng bộ mức tín hiệu 2 boar . Giữ xung nhịp đó và bạn vào set lại fuse bit cho chip bằng mạch ISP như bình thường ...  
  
Cách nạp lại chíp:  
  
Chúng ta cần tạo 1 nguồn xung cấp vào chân XTAL1 để kích nó hoạt động trở lại. Và cách thì vô số:  
  
**Cách 1: Dùng arduino.**  
  
Sẽ rất tiện nếu bạn có trong tay 1 board arduino, chỉ cần mở example blink ra, xóa delay đi và nạp thôi.  
  
Mã:  
  
void setup() {  
// initialize digital pin 13 as an output.  
pinMode(13, OUTPUT);  
}  
  
void loop() {  
digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)  
//delay(1000); // wait for a second  
digitalWrite(13, LOW); // turn the LED off by making the voltage LOW  
//delay(1000); // wait for a second  
}  
// ok  
Giờ ta chỉ việc nối chân 13 của arduino vào XTAL1 và khi ta đọc, ghi con AVR, lỗi ban nãy đã tan đi như làn khói mỏng.

**cách 2: Dùng IC 555.**  
  
Không phải lúc nào cũng sẵn có 1 con MCU như vậy, đơn giản hơn ta có thể dùng 1 con 555, vài cái dây nối, thế là ổn.  
  
[Diagram

Description automatically generated](https://2.bp.blogspot.com/-cSpDacRVm5Q/XZOA_u1WE-I/AAAAAAAAlx0/gogynonT2iAzvTmgtNE-REsBiVU_a-aVACK4BGAYYCw/s1600/555.png)  
  
Chẳng cần đến 1 cái tụ điện hay điện trở, thậm chí không cần nối 8 với 4. Giờ chỉ cần nối chân 3 của 555 vào XTAL1 là xong.  
  
Mạch này sẽ dao động với tần số cao nhất có thể của 555, phụ thuộc vào chất lượng của 555, bản CMOS sẽ nhanh hơn bản TTL.  
  
**Cách 3: Mạch đa hài.**  
  
Lắp lấy 1 cái mạch đa hài có tần số >10Khz và đấu output ra ngoài cũng cứu được. Tương tự nhé.  
  
Họ nói xung cấp vào nên là xung vuông độ rộng đều nhau ít nhất là 1Mhz nhưng sau khi phá phách, thử nghiệm tôi thấy chỉ cần là xung >10Khz, bất kể độ rộng bao nhiêu cũng được, thậm chí là xung sine. VD như mạch 555 ở trên có low time cực ngắn so với high time nhưng vẫn chạy tốt.  
  
Hi vọng bài viết này giúp các bạn tiết kiệm công sức và tiền bạc, không phải lọ mọ chạy ra tiệm xài nhờ máy tạo xung hay vứt oan chip (đắt lòi ra). Giờ thì thoải mái phang phập, set lung tung lên cho biết. Lỗi lại lôi hàng ra =)) =.=  
 **Cách 4:** Dùng máy kiểm tra linh kiện đa năng của [linhkiencuatui.com](http://linhkiencuatui.com/)  
  
Các bạn có thể mua tại địa chỉ: [http://linhkiencuatui.com](http://linhkiencuatui.com/)  
Với giá **230k**  
  
Kẹp chân Xtal 1 và Xtal 2 với chân 1 và chân 2 của máy đo linh kiện hoặc chân 2 và chân 3 của máy đo linh kiện.  
  
Nhấn giữ nút test --> nhấn từng bước để di chuyển xuống F-generator --> để ở tần số 2MHz  
  
Các bạn nạp bình thường bằng progisP --> OK