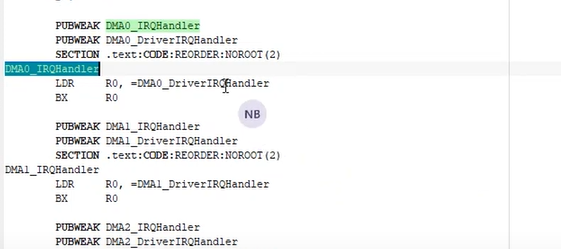
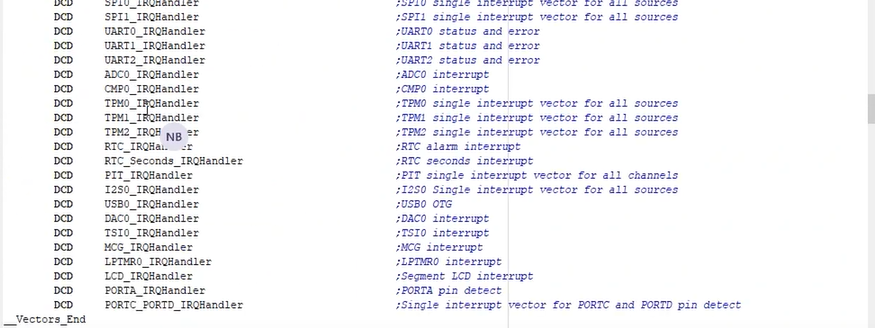
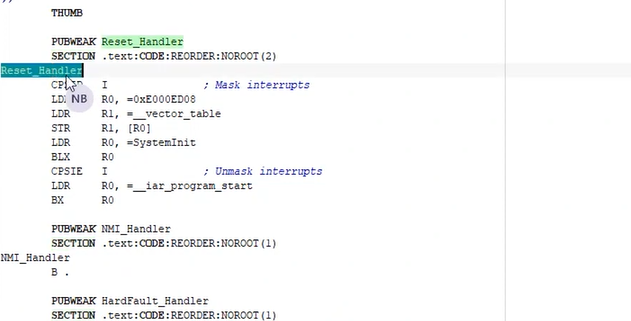


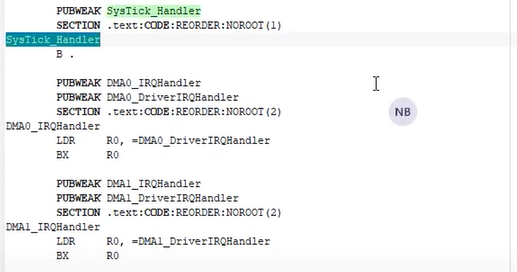
Nếu nhỡ tay set interrupt mà ko cài interrupt handler cho nó thì giá trị của bảng vector ban đầu ko được program thì sẽ có những giá trị linh tinh, dựa vào những giá trị đó nhảy lung tung khắp chip và dẫn tới treo, nhưng người ta ko muốn việc đó và người ta progam vector mặc định, thì khi nhảy vào nó sẽ, giả sử DMA chẳng hạncó địa chỉ của handler trong đó,



Trên này là địa chỉ của handler



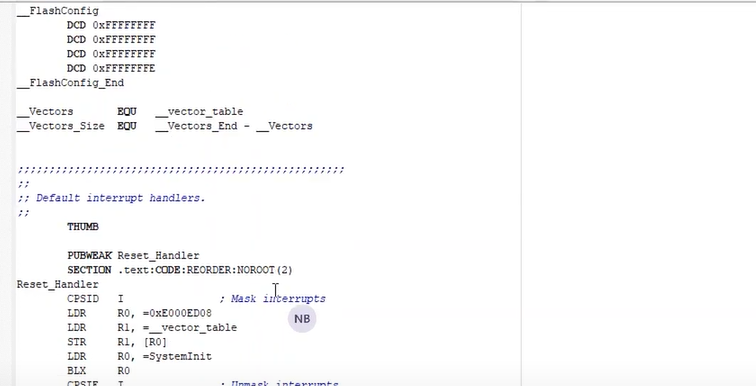
ở dưới là handler, nó là function



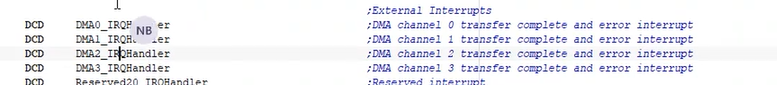
Ví dụ systick chẳng hạn, thử enable interrupt mà ko cài vector table interrupt cho nó chưa? Thì nó sẽ bị treo ở trong handler này (bôi xanh)

Câu lệnh B . tương đương với câu lệnh while (1), đó là lý do đèn sáng suốt vì chả có gì điều khiển cổng GPIO nữa, hoặc là tắt suốt.

Vậy làm sao để điều khiển bảng này



Khi bật vector interrupt lên thì hàm phục vụ ngắt của mình ở đây, cho nên bảng



Này tương ứng với thằng systick



Thì giả sử hàm của mình ko phải systick handler mà là systick isr chẳng hạn thì cần phải điền tên của hàm đó vào đây

Công việc này có thể do complier làm

Cụ thể là systick handler mặc định của file này, còn khi định nghĩa hàm của mình, systick iar thì ko được

Phải lập trình được bảng interrupt này để nó ghi vào trong phần đầu của bộ nhớ Flash, khi gặp interrupt thì nó sẽ tham chiếu tới interrupt ID bao nhiêu thì nó sẽ tương ứng với địa chỉ ở đây

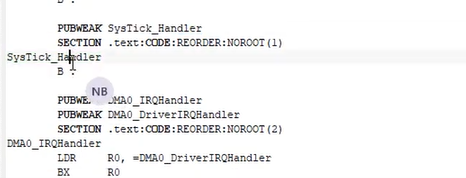
 ghi địa chỉ của hàm đó vào đây

Địa chỉ ghi dưới mã asembly mà thôi, viết bằng ngôn ngữ C thì viết kiểu khác

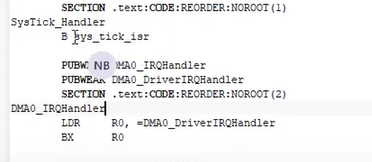


Giả sử hàm của mình là systick isr này thì địa chỉ của nó sẽ ở đây bằng cách ghi tên hàm của nó ở đây. Thì đó là 1 cách.

Cách thứ 2 là ở trong hàm hanlder này gọi hàm của mình



Thì nó sẽ nhảy bằng cách sửa dấu chấm này bằng systick isr



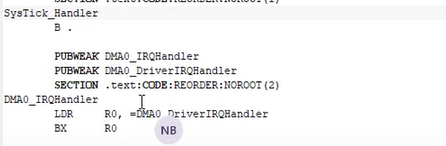
Đây là câu lệnh nhảy nên nó sẽ nhảy vào symbol nhất định, tuy nhiên nếu systick handler ở xa vị trí systick handler quá thì ko nhảy được, lúc đó sửa thành BX



Cách thứ 3, cách mà ngta hay sử dụng là ngta viết 1 thành 1 hàm trong file C tên giống y hệt cái này



Chuyện gì sẽ xảy ra, viết như thế có bị conflict với những thằng ở dưới ko



Mình có thể xoá, nhưng mà mình ko muốn xoá.

Câu trả lời khác: có thể có 2 symbol giống hệt nhau sẽ có conflict, nó như kiểu 2 biến global giống nhau conflict. Lúc link với nhau nó sẽ phát hiện ra các symbol trùng lặp, lỗi ngay ở quá trình link

Vậy nên có 1 từ khoá đặc biệt

PUBWEAK

Từ khoá này cho phép ghi đè, tức là có thằng nào define giống nó thì nó sẽ ghi đè, ko hẳn là ghi đè mà symbol này sẽ biến mất



Nó overwrite luôn

Người dùng khi sử dụng đến thư viện thì người ta ko muốn sờ đến file asm này vì khó hiểu cho nên ngta lựa chọn viết hàm handler giống như vậy

Đối với 2 cách đầu tiên, có 1 vấn đề nhỏ là đối với function này

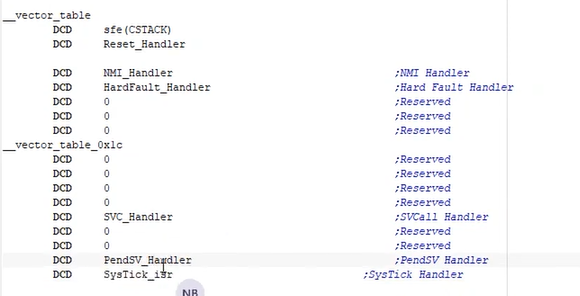


Thì nó đang ko nhận ra symbol này là gì

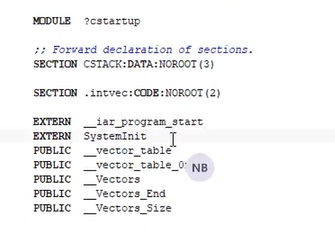


Nó sẽ báo lỗi là unresult symbol tại vì mình chưa declaration của nó

File.s này cũng giống file.c là nó cần hiểu những symbol này nghĩa là gì



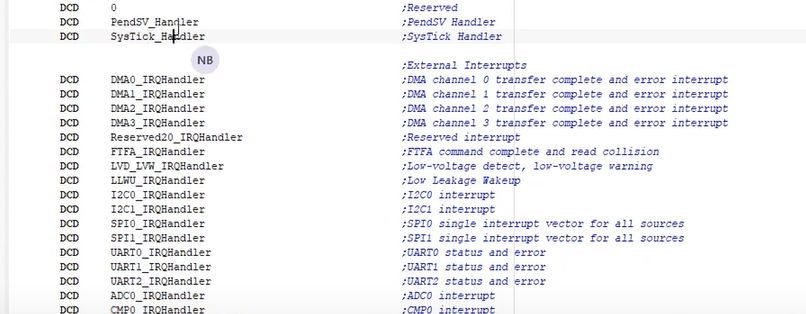
Cho nên cần phải extern nó (symbol ở ngoài) vào



Câu hỏi: khi xảy ra ngắt thì nó chạy vào bảng vector ngắt này để nó tìm cái handler của mình sau đó đi đến handler của mình phải ko?

Có thể có và có thể ko, trường hợp ko là nó sẽ có 1 phép init riêng để nó tự nhảy ngay lập tức đến địa chỉ kia luôn mà ko nhảy đến bảng vector ngắt này, tuỳ vào 1 số dòng vi xử lý có đưa con trỏ PC đến bảng vector ngắt này sau đó nhảy đến bảng hanlder này của mình, tuy nhiên các dòng nó có những phép get init đôi thì nó sẽ get những thằng này trong lúc nó đang thực hiện các thứ ở hàm main thì nó sẽ khác để nó cách những thằng này, tức là ngay lập tức nó đưa con trỏ PC sang thằng kia luôn chứ ko nhảy vào đây. Tuỳ vào thiết kế

Ví dụ ngắt của systick là nó có ID, ví dụ systick được nối với dây thứ 5 trong con chip chẳng hạn

thì mình có thể coi nó là dây thứ mười mấy ở đây từ 1 hay 10 đến 15 chẳng hạn

Tín hiệu của nó gửi vào dây thứ 15 thì nó sẽ báo bus nào phải fetch giá trị thứ 15 này chẳng hạn (tương ứng với từng dòng) và nó sẽ đưa con trỏ PC tới thằng này (vector thứ 15 này)

Vị trí của nó là độc nhất

ứng với mỗi interrupt thì nó chỉ có mỗi 1 vector ngắt, 1 cái interrupt handler duy nhất, muốn thay đổi cái đó thì phải program bảng interrupt này nhưng mà bảng interrupt này hiện tại đang nằm trên flash, muốn thay đổi bảng interrupt thì phải đưa nó lên RAM