STM8 được thiết kế với nhiều bộ Timer/Counter theo từng chip : Có chức năng định thời, đếm sự kiện, tạo xung điều rộng PWM, căn chỉnh thời gian như một bộ RTC. Ta sẽ xét từng bộ timer một xem sự khác nhau và giống nhau của chúng để các bạn có thể chọn bộ timer tương thích với ứng dụng của bạn:  
Ta sẽ thay định nghĩa Timer/Counter gọi chung là Timer.

1. Timer 4  
Đây là một bộ định thời có độ dài 8bit, nó hoạt động tương tự như chế độ Timer 8bit của AVR nhưng nó sẽ đếm từ 0 cho tới TOP ( được thiết lập sẵn ), giá trị tối đa của TOP là 2^8-1 tức là 255, khi đó sẽ có 1 ngắt xảy ra.  
Các thanh ghi phụ trách điều khiển Timer 4   
( Nên tham khảo thêm [datasheet STM8](http://http/shink.in/1Io2O) để dễ hiểu hơn )

1. TIM4\_CR1 : Thanh ghi chọn chế độ cho Timer 4. Ta chỉ cần quan tâm tới bit 0 - CEN ( cho phép Timer hoạt  động ) và bit 7 - ARPE ( chế độ tự nạp lại )
2. TIM4\_IER : Thanh ghi cho phép ngắt khi Timer 4 đếm tới TOP. Ta chỉ cần quan tâm tới bit 0 - UIE ( cho phép ngắt hay không )
3. TIM4\_SR : Thanh ghi chứa các trạng thái của các cờ. Ta quan tâm tới bit 0 - UIF ( cờ ngắt tràn )
4. TIM4\_CNTR : Thanh ghi chứa giá trị của Timer, mỗi khi có một xung theo hệ số chia, CNTR sẽ đếm lên 1
5. TIM4\_PSCR : Thanh ghi chọn hệ số chia lấy từ Fmaster. Ta sẽ có Ftimer = Fmaster/(2^PSCR)
6. TIM4\_ARR : Thanh ghi chứa giá trị Auto Reload đồng thời là TOP của Timer. Khi CNTR tăng đến và bằng giá trị trong ARR thì nó sẽ đếm lại về 0

Có 2 điểm cần lưu ý quan trọng

1. Nếu sử dụng chế độ auto reload thì thanh ghi ARR không được gán = 0
2. Phải xóa cờ trước khi thoát ngắt, nếu không xóa cờ sẽ không thoát được ngắt

TIM4\_SR\_UIF = 0;

TIM4\_SR = 0x00; *// Xóa hết cờ*

TIM4\_SR\_UIF = 0; có thể thay thế bằng TIM4\_SR &= 0xFE;

Ví dụ :   
Giả sử giờ muốn mỗi lần Timer đến nên 1 đơn vị mất 1us với Fmaster đang chạy max tốc độ là 16MHz  
Thì cần đặt tần số vào cho Timer là 1MHz => Hệ số chia là 16 => Ftimer = 16/16 = 16/2^4  
=> TIM4\_PSCR = 4;  
Để sử dụng chế độ Auto Reload thì phải Set bit 7 - ARPE trong CR1 thành 1 => TIM4\_CR1 = 0x81;  
  
Code ví dụ sử dụng Timer4 chế độ tự nạp lại và đếm từ 0 cho tới 100 thì xảy ra ngắt thay đổi trạng thái LED

TIM4\_CR1 = 0x81; *// thiết lập timer4 chế độ nạp lại*

TIM4\_PSCR = 4; *// hệ số chia là 4*

TIM4\_ARR = 100; *// TOP counter là 100*

TIM4\_IER = 0x01; *// cho phép ngắt*

#pragma vector = TIM4\_OVR\_UIF\_vector

\_\_interrupt void ISR\_TIM4(){

LED = ~LED;

TIM4\_SR\_UIF = 0;

}

2. Timer 1 và Timer 2  
Cũng tương tự như Timer 4 nhưng là Timer 16 bit nên Thanh ghi ARR và CNTR nó có 2 thanh ghi ARRH, ARRLvà CNTRH, CNTRL và có nhiều cờ hơn nên thanh ghi SR sẽ là SR1  
Timer 2 có tới 4 bit chọn hệ số chia tần trong thanh ghi PSCR nên nó chia đc cho 32768 là hệ số lớn nhất  
Còn hệ số chia tần của Timer 1 nó lại tính theo cách khác dùng tới 16 bit để chọn hệ số  
Ftimer1 = Fmaster / (TIM1\_PSCR + 1)