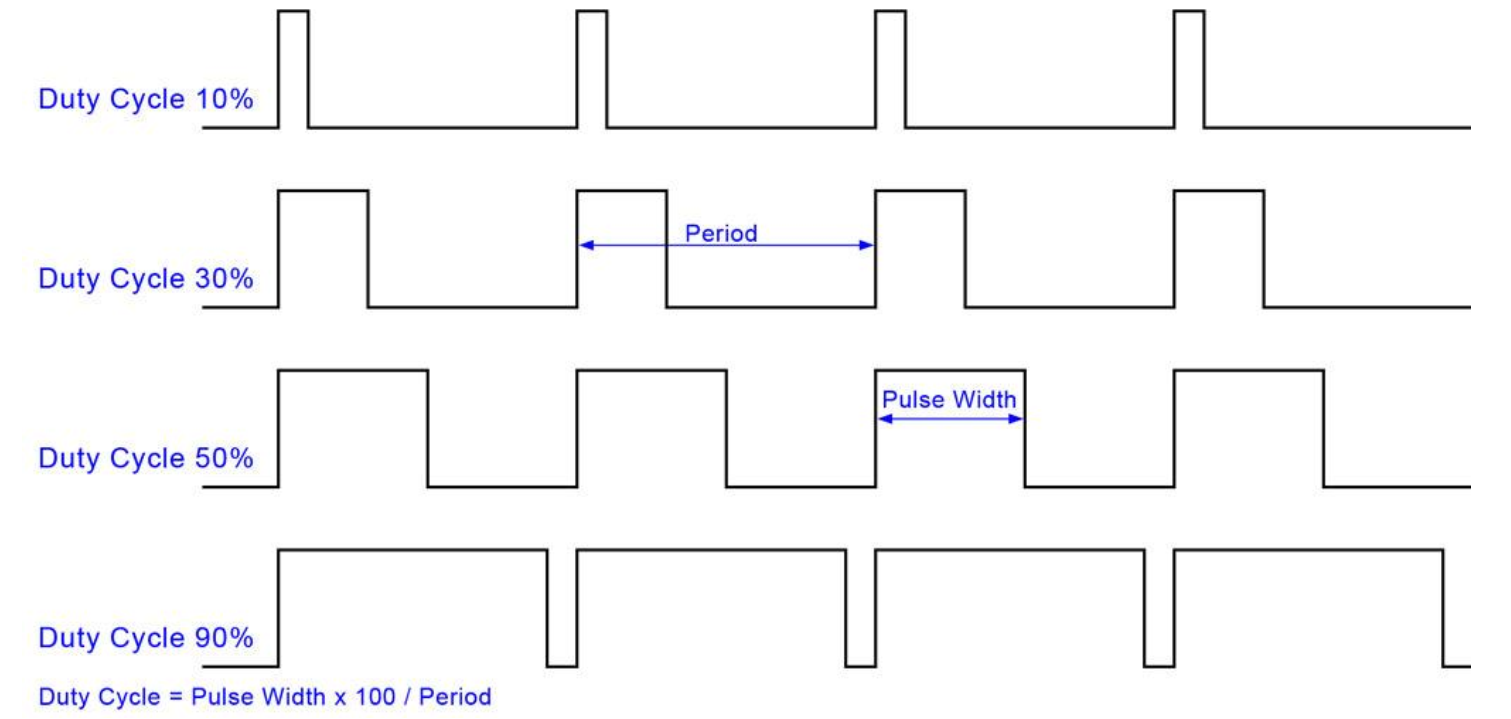
**Bài 17: PWM với STM32**

**Giới thiệu**

PWM (Pulse Width Modulation) – Điều chế độ rộng xung/Băm xung: là phương pháp điều chỉnh giá trị điện áp trung bình ra tải như các thiết bị như động cơ, đèn LED,… từ đó có thể làm thay đổi công suất thiết bị (tốc độ động cơ, độ sáng của đèn,…). Điều này được thực hiện bằng cách thay đổi Duty Cycle của xung tín hiệu điện áp ra, là phương pháp dễ dàng và ít tốn kém hơn việc điều chỉnh các thông số của dòng điện.

**I, Lý thuyết**

**-** Để hiểu đơn giản PWM là gì, bạn hãy tưởng tượng ta cần thay đổi độ sáng của một đèn LED. Khi đó, nếu ta cho LED sáng tắt liên tục với tần số cao, mắt chúng ta sẽ mất khả năng phân tích sự sáng tối của LED(do hiện tượng lưu ảnh), từ đó thấy được độ sáng khác nhau của đèn. Đèn sáng tỏ tức là thời gian sáng của đèn cao, đèn sáng mờ tức là thời gian sáng của đèn thấp.



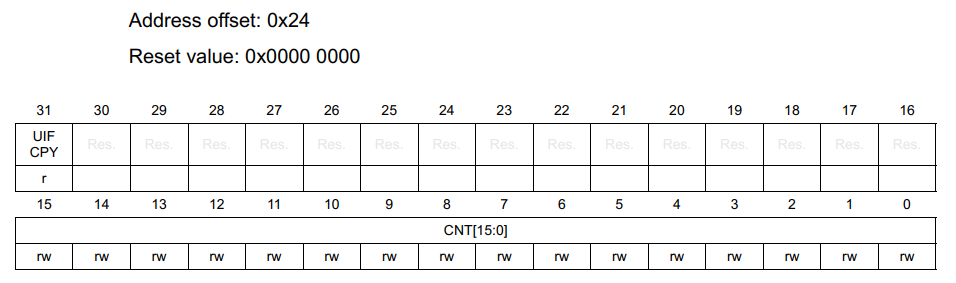
* + **Duty Cycle** là tỷ lệ phần trăm mức cao.
  + **Period** là chu kỳ xung.
  + **Pulse Width** là thời gian mức cao trong một chu kì.

**× 100%**

Khi làm việc với STM32, tính năng PWM nằm trong khối Timer của vi điều khiển.

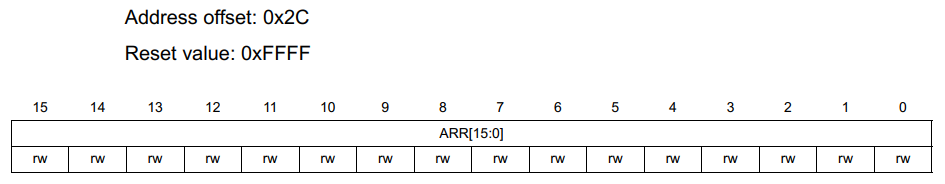
**Một số thanh ghi quan trọng trong chế độ PWM Generation:**

**TIMx counter register(TIMx\_CNT):** Thanh ghi lưu giá trị của Counter (CNT).



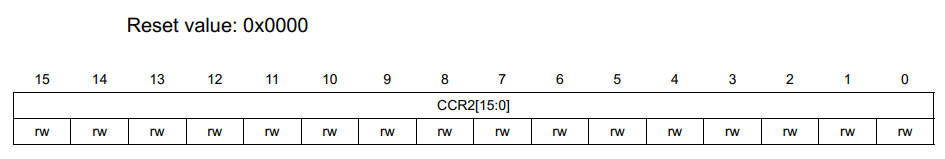
**CNT[15:0]** (Bit 15:0): Dãy bit lưu giá trị CNT.

**TIMx auto-reload register (TIMx\_ARR):** Trong chế độ PWM, thanh ghi này lưu giá trị độ phân giải của xung PWM (ví dụ độ phân giải là 100 thì LED có thể có 100 mức độ sáng khác nhau)



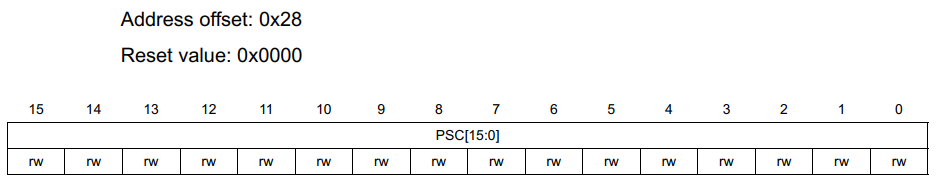
**ARR[15:0]** (Bit 15:0): Dãy bit lưu giá trị ARR

**TIMx capture/compare register x (TIMx\_CCRx):** Giá trị của thanh ghi này được dùng để làm mốc so sánh với Counter (sẽ được nói thêm ở phần dưới).



**CCR[15:0]** (Bit 15:0): Dãy bit lưu giá trị CCR

**TIMx prescaler (TIMx\_PSC):** Thanh ghi chứa giá trị chia xung clock được cấp vào từ bus APB.



**PSC[15:0]** (Bit 15:0): Dãy bit lưu giá trị chia xung PSC.

**Vi điều khiển STM32 có hỗ trợ 2 chế độ tạo xung PWM như sau:**

**Mode 1**

Nếu sử dụng chế độ đếm lên thì ngõ ra sẽ ở mức logic 1 khi CNT < CRR và ở mức 0 nếu CNT > CRR.

**Mode 2**

Nếu sử dụng chế độ đếm lên thì ngõ ra sẽ ở mức logic 0 khi CNT < CRR và ở mức 1 nếu CNT > CRR.

**II, Lập trình**

**Ý tưởng demo**: Mình sẽ sử dụng 4 kênh PWM của TIM4 trùng với 4 con LED trên KIT STM32F411VE của mình nhé. Sau đó sẽ xuất 4 xung PWM có duty cycle tăng dần làm và quan sát 4 LED trên KIT.

Đầu tiên cấu hình chế độ Debug Serial Wire

A screenshot of a computer

Description automatically generated

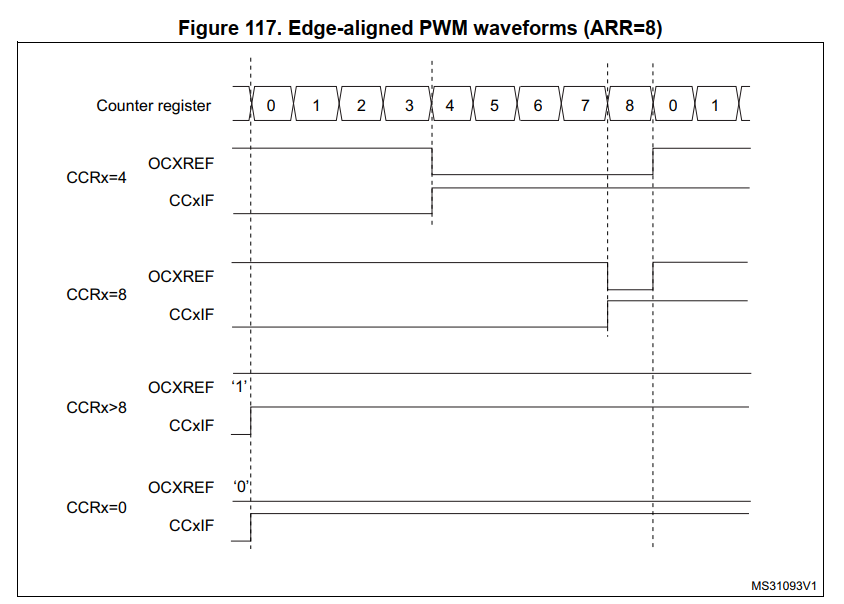
Tiếp theo mình cấu hình 4 Chanel PWM trên TIM4

A screenshot of a social media post

Description automatically generated

Ở đây mình cài đặt Timer ở chế độ CountingUp có Period đếm từ 0 → 99.

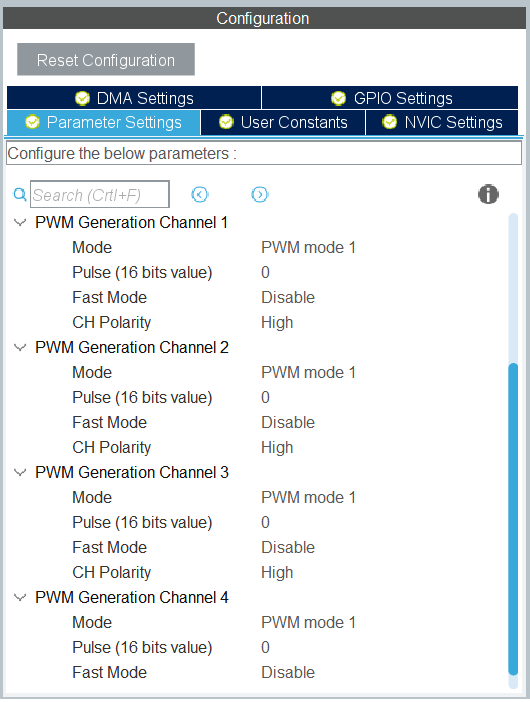
Các bạn theo dõi hình sau:



Như vậy

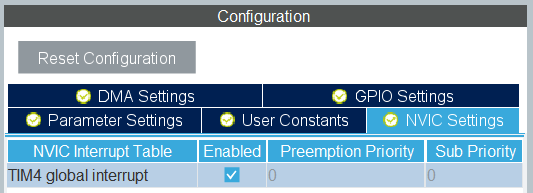
* duty cycle = 0 → CCRx = 0
* duty cycle = 1 → CCRx = 1
* duty cycle = 99 → CCRx = 99
* duty cycle = 100 → CCRx = 100

Tiếp theo cài đặt thông số 4 Channel PWM

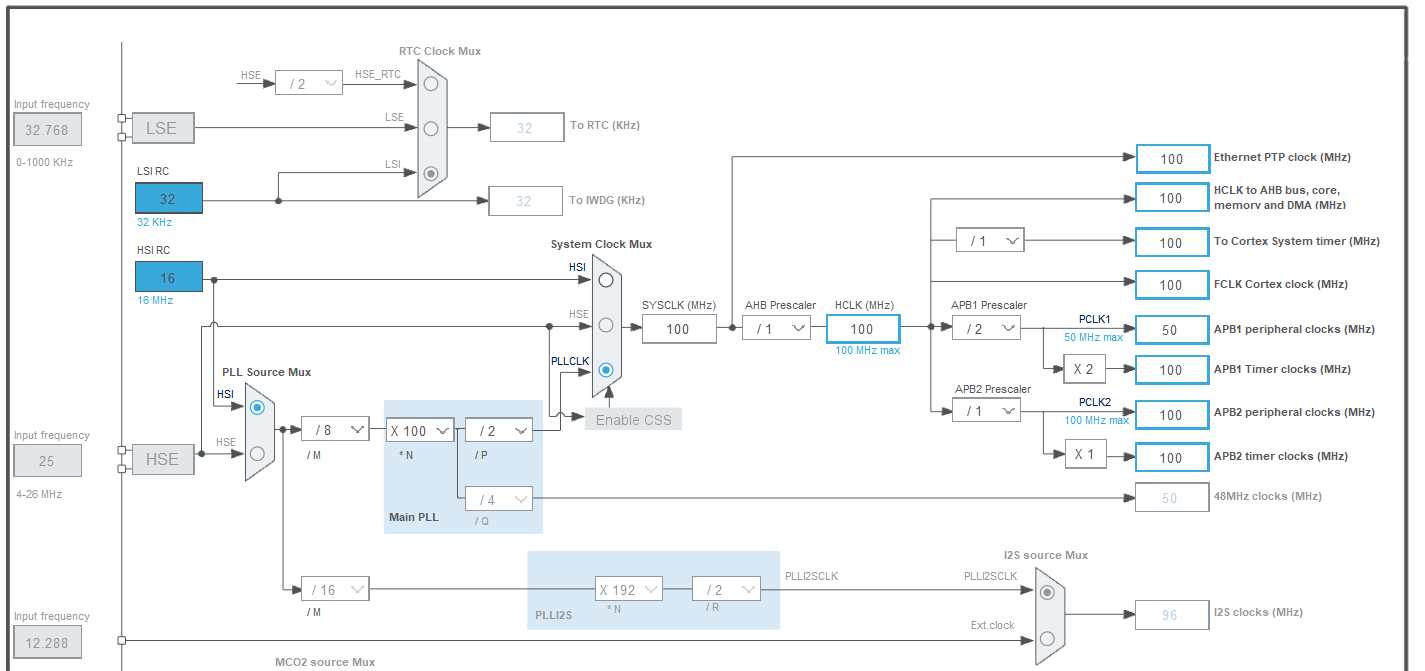


Bạn chọn CH Polarity là High, như vậy khi TIMxCNT < TIMxCCRx thì xung xuất ra ở mức “1”.

Enable ngắt cho TIM4.



Tiếp theo chúng ta sang cài đặt Clock cho hệ thống, ở đây mình sẽ sử dụng nguồn Clock nội có sẵn trong Chip.



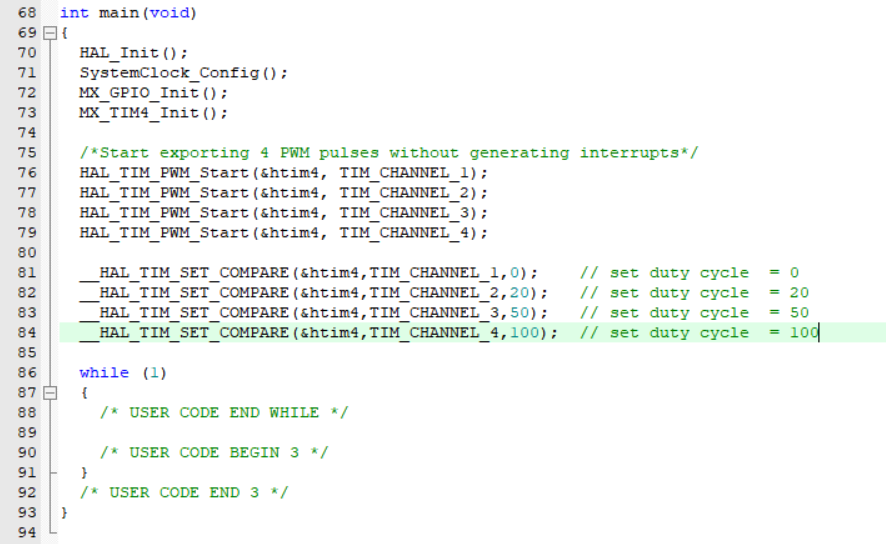
Nhìn vào đây có thể thấy bộ Timer được cung cấp nguồn Clock 100Mhz.

Chúng ta cấu hình tên dự án và để Cube MX sinh souce code.

A screenshot of a social media post

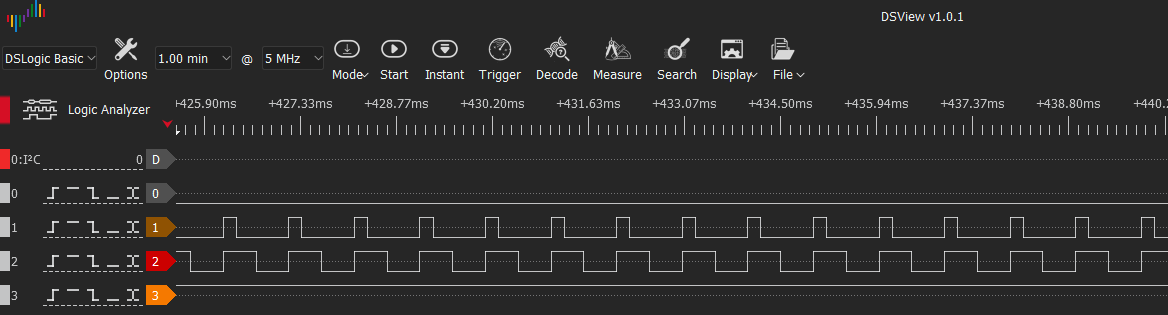
Description automatically generated

Nào giờ hãy cùng nhau bắt tay vào sửa code nhé.



Sau đó build code và nạp lên KIT.

Dưới đây là xung xuất ra trên 4 kênh PWM.



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DEVIOT - CÙNG NHAU HỌC LẬP TRÌNH IOT

📌 Website: deviot.vn

📌 FanPage: Deviot - Thời sự kỹ thuật & IoT

📌 Group: Deviot - Cùng nhau học lập trình IOT

📌 Hotline: 0969.666.522

📌 Address: Số 101C, Xã Đàn 2

📌 Đào tạo thật, học thật, làm thật