Giới thiệu Timer

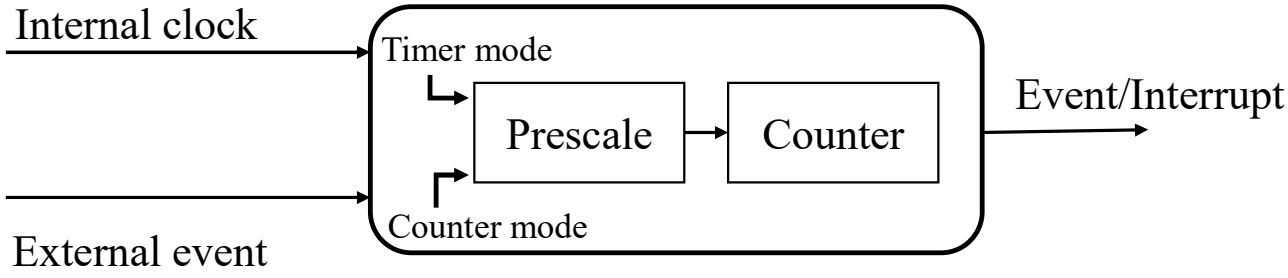
Sơ đồ khối cơ bản của 1 timer

Một timer cơ bản thông thường có 2 độ chính.

1. Chế độ định thời (timer mode)

2. Chế độ đếm xung ngoài (counter mode)

Basic Timer Block

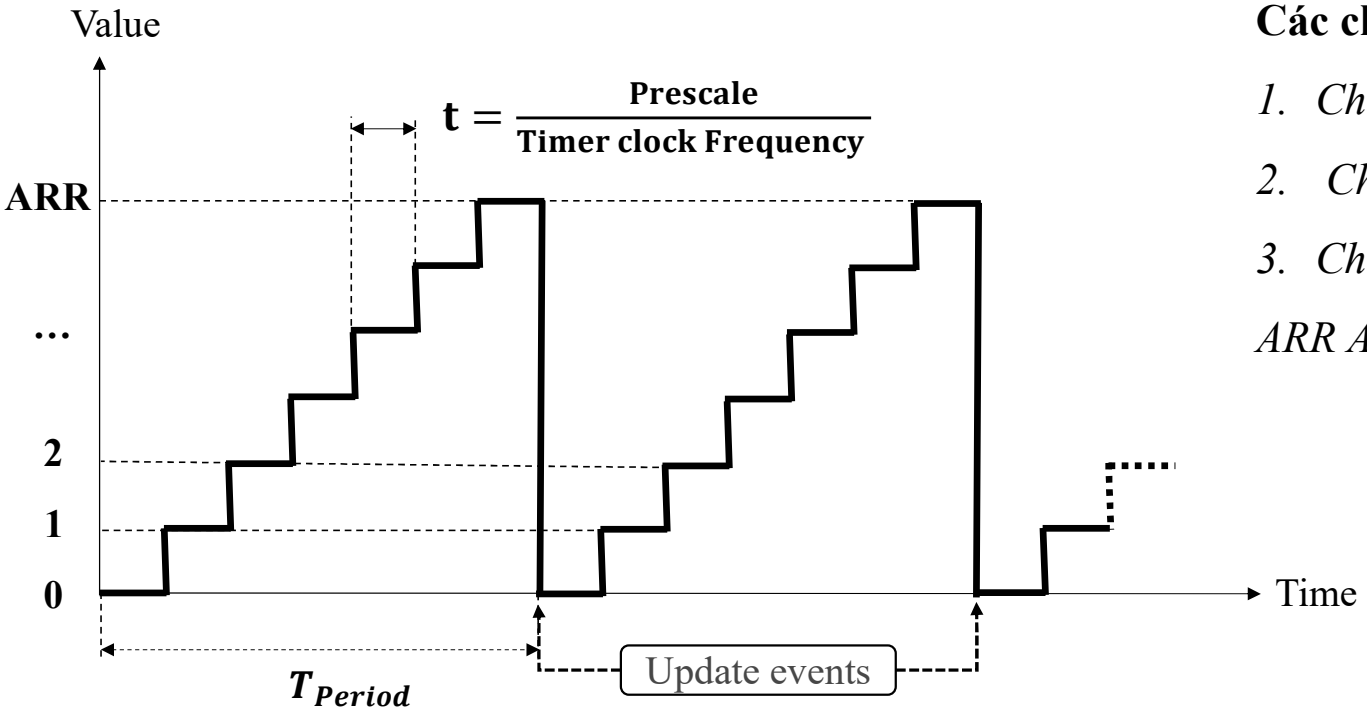


Prescale: Bộ chia tín hiệu xung

Internal clock: xung clock được cấp từ hệ thống clock của vi điều khiển

External event: xung clock/sự kiện được cấp vào chân IO tương ứng của timer

Giới thiệu Timer trong STM32



Ví dụ bộ đếm hoạt động với chế độ đếm lên

Các chế độ đếm

1. *Chế độ đếm lên 16bit (or 32 bit)*
2. *Chế độ đếm xuống 16bit (or 32 bit)*
3. *Chế độ vừa đếm lên, vừa đếm xuống*

ARR Auto Reload Register

Cấu hình cho Timer

Các bước thiết lập chế độ Timer mode cơ bản

Step 1: Cấp xung clock cho bộ Timerx (RCC register)

Step 2. Thiết lập bộ chia (Prescale) và giá trị đặt trước (ARR)

Step 3: Kích hoạt Timer

Step 4: Thiết lập Timer Interrupt (nếu sử dụng)

Cấu hình cho Timer

Step 1. Cấp xung hoạt động cho module Timerx (x: 1, 2, 3, 4,…)

Được quản lý bởi module RCC (Reset and Clock Control). Mỗi một module Timer nếu muốn hoạt động được thì cần phải được cấp xung (clock) từ nguồn xung của hệ thống.

- ***F1 Series: APB1ENR***

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DAC EN	PWR EN	BKP EN	CAN2 EN	CAN1 EN	Reserved		I2C2 EN	I2C1 EN	UART5EN	UART4EN	USART3 EN	USART2 EN	Res.
		rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Reserved		WWD GEN	Reserved					TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw			rw						rw	rw	rw	rw	rw	rw

- ***F2,3,4 Series: APB1ENR***

Ví dụ:

```
RCC -> APB1ENR |= (1<<0) ; // Ennable Clock for Timer 2
```

Step 2. Thiết lập bộ chia (PSC) và giá trị đặt trước (ARR)

Chu kỳ định thời của timer được quyết định bởi 2 thanh ghi: TIMx_PSC và TIMx_ARR

TIMx_PSC: Thanh ghi 16 bit cho phép thiết lập bộ chia cho timer tương ứng. Giá trị thiết lập từ 0 đến 65535

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Lưu ý: Giá trị thực của bộ chia được hiểu là PSC + 1

TIMx_ARR: Thanh ghi 16 (or 32) bit cho phép thiết lập giá trị đặt trước cho bộ đếm của timer tương ứng. Giá trị thiết lập từ 0 đến 65535

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16] (depending on timers)															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

TIMx_CNT: Thanh ghi làm nhiệm vụ đếm của timer

Step 2. Thiết lập bộ chia (PSC) và giá trị đặt trước (ARR)

Từ giá trị định thời, giá trị bộ chia và giá trị đặt trước được tính toán theo công thức sau.

$$Thời\ gian\ định\ thời\ Tout\ (s) \rightarrow Frequency\ of\ Tout(Hz) = \frac{Timer\ Clock\ Frequency\ (Hz)}{(Prescaler + 1)(ARR + 1)}$$

$$\Rightarrow \boxed{(Prescaler + 1)(ARR + 1) = Timer\ Clock\ Frequency\ (Hz) * Tout\ (s)}$$

Ví dụ: ***Cần định thời 1 khoảng thời gian: 100ms, Timer clock frequency: 8MHz***

$$(Prescaler + 1)(ARR + 1) = 8{,}000{,}000 * 0.1 = 800{,}000$$

1. ***Chọn: PSC = 79 => ARR +1 = 10,000***
2. ***Chọn: PSC = 799 => ARR +1 = 1,000***
3. ***…***

```
TIM2 -> PSC  = 79 ;
TIM2 -> ARR  = 10000 ;
```

Step 3. Kích hoạt Timer

Sau khi các thông số cần thiết cho timer đã hoàn thành, tiến hành kich hoạt timer ở thanh ghi TIMx_CR1

TIMx_CR1: Thanh ghi điều khiển các chế độ làm việc chung của timer tương ứng

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Ví dụ:

```
TIM2 -> CR1 |= (1<<0) ;
```

Lưu ý: Các giá trị bộ chia PSC chỉ được nạp cho vi điều khiển ngay sau khi bộ đếm của timer tràn lần đầu tiên

Step 3. Kích hoạt Timer

Các thanh ghi option: TIMx_EGR, TIMx_SR

TIMx_EGR: Thanh ghi khởi tạo các sự kiện cho timer

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

Ví dụ:

```
TIM2 -> EGR |= (1<<0) ;   // Generates and update of registers
```

TIMx_SR: Thanh ghi cho phép theo dõi các trạng thái của time

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Reserved		TIF	Res	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Step 4. Thiết lập Timer Interrupt

Mỗi khi bộ đếm của Timer bị tràn thì timer sẽ cho phép 1 sự kiện ngắt timer (nếu như nó được cấu hình).

TIMx_DIER: Thanh ghi cho phép kích hoạt các sự kiện ngắt/DMA của timer tương ứng

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

Ví dụ:

```
TIM2 -> DIER |= (1<<0) ;   // Update Interrupt Enable
```

Khởi tạo ngắt của hệ thống NVIC cho Timer

```
NVIC -> ISER[0] |= (1<<28) ;   // Timer 2 Interrupt Enable
```

Step 4. Thiết lập Timer Interrupt

Mỗi khi bộ đếm của Timer bị tràn thì con trỏ chương trình nhảy vào vector ngắt của timer để thực hiện

Vector ngắt của Timer.

```
/// timerx interrupt vector
void TIMx_IRQHandler (void)        // x = 1, 2, 3, 4,…..
{
        TIMx->SR &= ~(1<< 0);      //clear timerx interrupt flag
        //placed code begin

        //placed code end
}
```

Bài tập cơ bản

UTE

Bài tập 1 Viết chương trình thực hiện định thời chớp tắt led đơn với chi kỳ 100ms B10

Bài tập 2 Viết chương trình thực hiện định thời chớp tắt led đơn với chi kỳ 5000ms

Bài tập 3 Viết chương trình thực hiện định thời theo yêu cầu sau:

1. Nhấn Start (nút xanh) băng tải quay.
2. Sau đó, nhấn nút màu vàng cho băng tải dừng.
3. Nếu như sau 5 giây (5000ms) không nhấn nút vàng còn lại thì băng tải quay lại, trong thời gian đó, đèn xanh lá cây chớp tắt với chu ky 200ms. Nếu như nút vàng còn lại dược nhấn thì băng tải không quay.
4. Nếu như băng tải đang hoạt động, nút đỏ dc nhân thì bằng tải dừng lại tập tức.