

# ELF52 - Sistemas Microcontrolados

## Temporizadores

**Professor:**

Prof. Marcos Eduardo

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

# Temporizadores

# Introdução

- Como fazer contagem de tempo?
  - 3 formas

# Introdução

- Como fazer contagem de tempo?
  - 3 formas

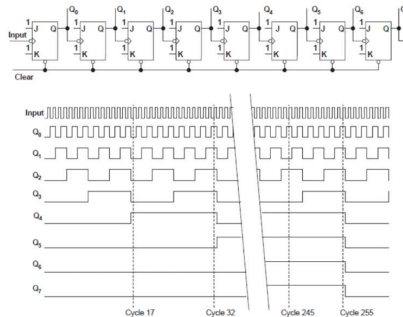


# Introdução

- Como fazer contagem de tempo?
  - 3 formas
    - Contagem por laço;
    - Contagem com *SysTick*;
    - Periférico.

# Introdução

- O que são temporizadores (*timers*)?
  - Periféricos empregados na geração periódica de pedidos de interrupção;
  - São arranjos de *flip-flops*.

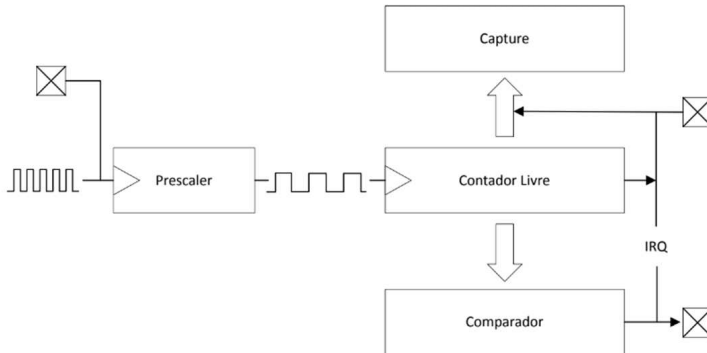


# Introdução

- Principais aplicações:
  - Contagem de tempo;
  - Contagem de eventos externos;
  - Contagem de largura de pulso;
  - Geração de onda quadrada e PWM.

# Visão Geral

- Diagrama de um *timer* genérico:







## Modos de operação

- *One-shot timer* programável de 16 ou 32 bits;
- *Timer* periódico de 16 ou 32 bits;
- Modo de captura: Contador de bordas de 16 bits;
- Modo de captura: Contagem **entre** bordas de 16 bits;
- O modo de 16 bits pode utilizar 8 bits de *prescaler*.

# Modos de operação

- *One-shot timer* → **Configura o timer para gerar uma interrupção temporizada;**
- *Timer periódico* → **Configura o timer para gerar uma interrupção periódica;**
- *Capture Mode: Input Edge Count* (Contador de bordas) → Configura o *timer* para contar pulsos e avisar quando ele contou um certo número;
- *Capture Mode: Input Edge Time* (Contagem **entre** bordas) → Configura o *timer* para contar o tempo entre eventos.

# Registadores

- Para ativar cada um dos temporizadores , o clock do respectivo temporizador tem que ser ativado no registrador **RCGCTIMER**:

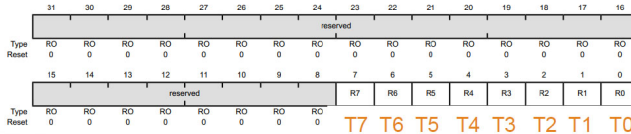
16/32-Bit General-Purpose Timer Run Mode Clock Gating Control (RCGCTIMER)

Base 0x400F.E000

Offset 0x504

Type RW, reset 0x0000.0000

**SYSTCTL\_RCGCTIMER\_R**



- Verificar o bit do temporizador respectivo no registrador **PRTIMER** para saber se está pronto para o uso:

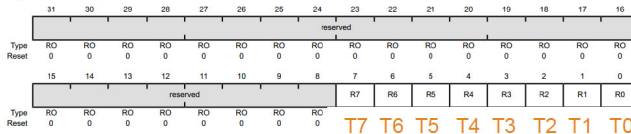
16/32-Bit General-Purpose Timer Run Mode Clock Gating Control (RCGCTIMER)

Base 0x400F.E000

Offset 0x504

Type RW, reset 0x0000.0000

**SYSTCTL\_RCGCTIMER\_R**

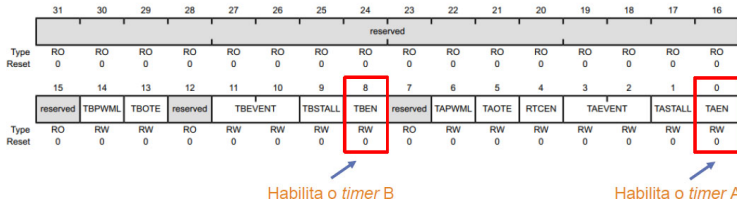


# Registadores

- O registrador **GPTMCTL** realiza o controle dos timers A e B, neste registrador que os timers são habilitados:

TIMERX\_CTL\_R

GPTM Control (GPTMCTL)

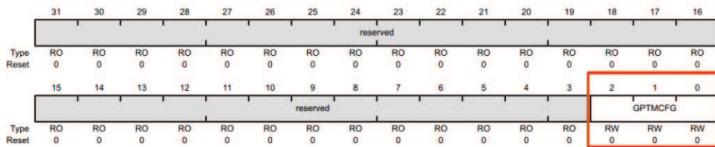


# Registadores

- O registrador **GPTMCFG** realiza a configuração de quantos bits será a contagem do temporizador:

TIMERX\_CFG\_R

GPTM Configuration (GPTMCFG)



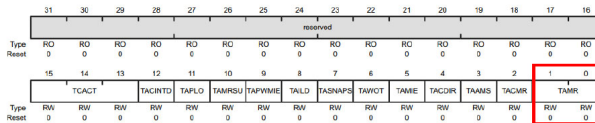
Contagem de 16 bits: 0x04  
Contagem de 32 bits: 0x00

# Registadores

- Os registradores **GPTMTAMR** e **GPTMTBMR** realizam a configuração do modo de operação dos timers A e B, respectivamente:

## TIMERX TAMR\_R

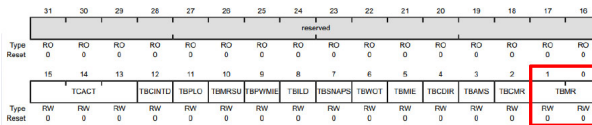
GPTM Timer A Mode (GPTMTAMR)



Modo do Timer A  
One Shot: 0x1  
Periódico: 0x2  
Captura: 0x3

## TIMERX\_TBMR\_R

GPTM Timer B Mode (GPTMTBMR)



Modo do Timer B  
One Shot: 0x1  
Periódico: 0x2  
Captura: 0x3

\*Para o modo de captura, o TnCMR especifica qual o modo (0 *Edge-Count*, 1 *Edge-Time*) específico.

# Registadores

- Os registradores **GPTMTAILR** e **GPTMTBILR** são os registradores de carga de contagem dos timers A e B, respectivamente. Se configurado no modo 32 bits, somente o **GPTMTAILR** é utilizado:

GPTM Timer A Interval Load (GPTMTAILR)

TIMERX\_TAILR\_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Valor da contagem calculado baseado no clock. Se 32 bits, preencher apenas este registrador.

GPTM Timer B Interval Load (GPTMTBILR)

TIMERX\_TBILR\_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

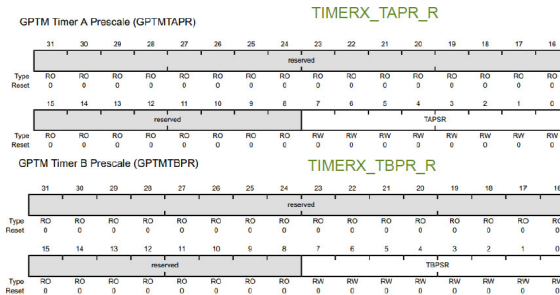
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Valor da contagem se o timer B for utilizado como 16 bits.



# Registadores

- Os registradores **GPTMTAPR** e **GPTMTBPR** são os registradores de preescala dos timers A e B, respectivamente se configurados como 16 bits. Se não desejar preescala, deixar zerado:



# Registadores

- O registrador **GPTMIMR** realiza o controle da interrupção timers A e B. Neste registrador a interrupção a nível de periférico é habilitada:

TIMERX\_IMR\_R

GPTM Interrupt Mask (GPTMIMR)

	31	30	29	28	27	26	25	24	reserved				23	22	21	20	19	18	17	16
Type	RO	RO	RO	RO	RO	RO	RO	RO					RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
	reserved		DMABIM	reserved	TBMIM	CBEIM	CBMIM	TBTOIM	reserved		DMAAIM	TAMIM	RTCIM	CAEIM	CAMIM	TATOIM				
Type	RO	RO	RW	RO	RW	RW	RW	RW	RO	RO	RW	RW	RW	RW	RW	RW				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Habilita a  
interrupção do  
timer B

Habilita a  
interrupção do  
timer A

# Registradores

- O registrador **GPTMICR** limpa a interrupção timers A e B. Por ele, é realizado o ACK da interrupção:

TIMERX\_ICR\_R

GPTM Interrupt Clear (GPTMICR)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16												
	reserved																											
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
	reserved		DMAIBINT		reserved		TBMCIINT		CBECINT		CBMCINT		TBTOCINT		reserved		DMAAINT		TAMCINT		RTCCINT		CAECINT		CAMCINT		TATOCINT	
Type	RO	RO	W1C	RO	W1C	W1C	W1C	W1C	W1C	RO	RO	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Limpa a  
interrupção do  
timer B

Limpa a  
interrupção do  
timer A

## Modos *One-Shot*

- O *timer* pode ser configurado para contar para cima ou para baixo;
- No modo de contagem para baixo se o contador chega em 0, o *timer* para e gera um evento de interrupção se configurada. O valor de início de contagem deve ser configurado;
- Na contagem para cima o *timer* para quando chegar no valor de *timeout*.

## Modos *One-Shot*

- O *clock* do *timer* é o **mesmo** do *clock* do sistema. Ou seja, para um *clock* de 80MHz cada contagem acontece a cada  $1/80\text{MHz}$   
 $= 12,5\text{ns}$ .
- Temos 8 *timers* disponíveis para uso;
- Cada *timer* pode ser dividido em **TimerA** e **TimerB**:
  - Quando utilizados os *timers* separados, os *timers* são de **16 bits**, possibilitando contagens de 0 a 65535;
  - Se unir os dois *timers* pode-se ter um *timer* com contagem até **32 bits**;
  - Há ainda um *prescaler* de **8 bits** para ser utilizado nos *timers* de **16 bits**, que atua como um divisor da frequência do *clock*.

## Modos *One-Shot*

- Tempo de contagem do *timer*:

$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$

- Contagem depende se o modo é 16 ou 32 bits;
- *Prescale* é limitado a 255(+1) e só é permitido no modo de 16 bits, aumentando o *timer* para até 24bits.

# Limites de Tempo

- Limite para 16 bits:  
 $16 \text{ bits} \Rightarrow 2^{16} \cdot \frac{1}{80 \cdot 10^6} = 0,8192ms$
- Limite para 16 bits+prescale:  
 $16 \text{ bits} \Rightarrow 2^{16+8} \cdot \frac{1}{80 \cdot 10^6} = 0,20971s$
- Limite para 32 bits:  
 $16 \text{ bits} \Rightarrow 2^{32} \cdot \frac{1}{80 \cdot 10^6} = 53,687s$

# Limites de Tempo

- Limite para 16 bits:  
 $16 \text{ bits} \Rightarrow 2^{16} \cdot \frac{1}{80 \cdot 10^6} = 0,8192ms$
- Limite para 16 bits+prescale:  
 $16 \text{ bits} \Rightarrow 2^{16+8} \cdot \frac{1}{80 \cdot 10^6} = 0,20971s$
- Limite para 32 bits:  
 $16 \text{ bits} \Rightarrow 2^{32} \cdot \frac{1}{80 \cdot 10^6} = 53,687s$



# Limites de Tempo

- Limite para 16 bits:  
 $16 \text{ bits} \Rightarrow 2^{16} \cdot \frac{1}{80 \cdot 10^6} = 0,8192ms$
- Limite para 16 bits+prescale:  
 $16 \text{ bits} \Rightarrow 2^{16+8} \cdot \frac{1}{80 \cdot 10^6} = 0,20971s$
- Limite para 32 bits:  
 $16 \text{ bits} \Rightarrow 2^{32} \cdot \frac{1}{80 \cdot 10^6} = 53,687s$

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 16 bits com prescale;
- Fixamos o prescale em 99 (poderia ser até 255):  
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:  
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x \cdot (99 + 1)$$
- De maneira que:  
$$x = \frac{80 \cdot 10^6}{100} \cdot 10ms = 8000$$
- Por fim,  $(contagem + 1) = 8000$ , de maneira que  $contagem = 7999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 16 bits com prescale;
- Fixamos o prescale em 99 (poderia ser até 255):  
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:  
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x \cdot (99 + 1)$$
- De maneira que:  
$$x = \frac{80 \cdot 10^6}{100} \cdot 10ms = 8000$$
- Por fim,  $(contagem + 1) = 8000$ , de maneira que  $contagem = 7999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 16 bits com prescale;

- Fixamos o prescale em 99 (poderia ser até 255):

$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$

- Assumindo que  $x = (contagem + 1)$ , temos que:

$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x \cdot (99 + 1)$$

- De maneira que:

$$x = \frac{80 \cdot 10^6}{100} \cdot 10ms = 8000$$

- Por fim,  $(contagem + 1) = 8000$ , de maneira que  $contagem = 7999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 16 bits com prescale;
- Fixamos o prescale em 99 (poderia ser até 255):  
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:  
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x \cdot (99 + 1)$$
- De maneira que:  
$$x = \frac{80 \cdot 10^6}{100} \cdot 10ms = 8000$$
- Por fim,  $(contagem + 1) = 8000$ , de maneira que  $contagem = 7999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 16 bits com prescale;
- Fixamos o prescale em 99 (poderia ser até 255):  
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1) \cdot (prescale + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:  
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x \cdot (99 + 1)$$
- De maneira que:  
$$x = \frac{80 \cdot 10^6}{100} \cdot 10ms = 8000$$
- Por fim,  $(contagem + 1) = 8000$ , de maneira que  $contagem = 7999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 32 bits;
- Lembrando que:
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x$$
- De maneira que:
$$x = 80 \cdot 10^6 \cdot 10ms = 800.000$$
- Por fim,  $(contagem + 1) = 800.000$ , de maneira que  $contagem = 799.999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 32 bits;

- Lembrando que:

$$Tempo = \left( \frac{1}{sysclk} \right) \cdot (contagem + 1)$$

- Assumindo que  $x = (contagem + 1)$ , temos que:

$$10ms = \left( \frac{1}{80 \cdot 10^6} \right) \cdot x$$

- De maneira que:

$$x = 80 \cdot 10^6 \cdot 10ms = 800.000$$

- Por fim,  $(contagem + 1) = 800.000$ , de maneira que  $contagem = 799.999$ .



## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 32 bits;

- Lembrando que:

$$Tempo = \left( \frac{1}{sysclk} \right) \cdot (contagem + 1)$$

- Assumindo que  $x = (contagem + 1)$ , temos que:

$$10ms = \left( \frac{1}{80 \cdot 10^6} \right) \cdot x$$

- De maneira que:

$$x = 80 \cdot 10^6 \cdot 10ms = 800.000$$

- Por fim,  $(contagem + 1) = 800.000$ , de maneira que  $contagem = 799.999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 32 bits;

- Lembrando que:

$$Tempo = \left( \frac{1}{sysclk} \right) \cdot (contagem + 1)$$

- Assumindo que  $x = (contagem + 1)$ , temos que:

$$10ms = \left( \frac{1}{80 \cdot 10^6} \right) \cdot x$$

- De maneira que:

$$x = 80 \cdot 10^6 \cdot 10ms = 800.000$$

- Por fim,  $(contagem + 1) = 800.000$ , de maneira que  $contagem = 799.999$ .

## Exemplo

- Assumindo que queremos uma contagem a cada 10ms utilizando 32 bits;
- Lembrando que:  
$$Tempo = \left(\frac{1}{sysclk}\right) \cdot (contagem + 1)$$
- Assumindo que  $x = (contagem + 1)$ , temos que:  
$$10ms = \left(\frac{1}{80 \cdot 10^6}\right) \cdot x$$
- De maneira que:  
$$x = 80 \cdot 10^6 \cdot 10ms = 800.000$$
- Por fim,  $(contagem + 1) = 800.000$ , de maneira que  
 $contagem = 799.999$ .

## Modos *One-Shot*

- Passo-a-passo para habilitar o modo *one-shot*:
  - 1 Habilite o respectivo *timer* no registrador **RCGCTIMER** (cada bit representa um *timer*) e esperar até que o respectivo *timer* esteja pronto para ser acessado no registrador **PRTIMER** (cada bit representa um *timer*);
  - 2 Garanta que o *timer* esteja desabilitado antes de fazer as alterações (limpar o bit **TnEN**, onde **n** pode ser **A** ou **B**) no registrador **GPTMCTL** (*Control*);
  - 3 Coloque o *timer* no modo 16 ou 32 bits escrevendo 0x4 ou 0x0 no registrador **GPTMCFG** (*Configuration Register*);
  - 4 Configure o campo **TnMR** para 0x1 no registrador **GPTMTnMR** (*Timer n Mode Register*).

## Modos *One-Shot*

- Passo-a-passo para habilitar o modo *one-shot* (cont):
  - 5 Carregue o valor de contagem desejado no registrador **GPTMTnILR**:
    - Se o *timer* for de 32 bits (unindo os dois *timers*) escrever o valor de 32 bits apenas no **GPTMTAILR**;
  - 6 Carregue o valor de *prescale* no **GPTMTnPR** (*Timer n Prescale*) se o *timer* for de 16 bits.

## Modos *One-Shot*

- Passo-a-passo para habilitar o modo *one-shot* (cont):
  - ⑦ Escreva 1 no bit **TnTOCINT** (*Timer n Time-Out Interrupt Clear*) para limpar o *flag* de interrupção no registrador **GPTMICR** (*Interrupt Clear Register*);
  - ⑧ Se interrupções são desejadas:
    - Ⓐ Escrever 1 no bit **TnTOIM** (*Timer n Time-Out Interrupt Mask*) para habilitar a interrupção no registrador **GPTMIMR** (*Interrupt Mask Register*);
    - Ⓑ Sete a **prioridade** da interrupção do *timer* respectivo no respectivo registrador **NVIC Priority Register**;
    - Ⓒ **Habilitar** a interrupção do *timer* respectivo no respectivo registrador **NVIC Interrupt Enable Register**.

## Modos *One-Shot*

- Passo-a-passo para habilitar o modo *one-shot* (cont):
  - 9) Habilite o bit **TnEN** no registrador **GPTMCTL** (*Control*) para começar o *timer* e iniciar a contagem de modo decrescente do valor da contagem;
  - 10) Se:
    - a) A interrupção estiver habilitada, a rotina de tratamento da interrupção (e.g. **TimerXn\_Handler**) deve tratar o evento, escrevendo 1 no bit **TnTOCINT** do registrador **GPTMICR** que limpa o *flag* de interrupção;
    - b) A interrupção não estiver habilitada, realizar *polling* do bit **TnTORIS** do registrador **GPTMRIS**. Lembre de limpar o *flag* de interrupção escrevendo 1 no bit **TnTOCINT** do registrador **GPTMICR**.

## Modos *Periódico*

- O modo periódico é muito parecido com o one-shot;
- A única diferença é que o *timer* **não para** depois de chegar a 0: quando ele chega a 0, ele reinicia com o valor carregado no **GPTMTnILR**;
- Para configurar o *timer* no modo periódico, mudar o **passo 4** do modo *one-shot*:
  - Configurar o campo **TnMR** para `0x2` no registrador **GPTMTnMR** (*Timer n Mode Register*).



# Outros Modos

# Modo de Captura: Modo Contador de Bordas

- Conta quantas bordas de subida, descida ou ambas acontecem no pino CCP e armazena o valor da contagem nos registradores **GPTMTnR**;
- Conta até 24 bits (utilizando o *prescaler* como os 8 bits superiores);
- O respectivo pino de entrada/saída deve ser configurado para função especial CCP conforme a aula de GPIO;
- Detalhes de configuração (ver *Datasheet* seção 13.4.3 na página 972).

## Modo de Captura: Contagem entre bordas

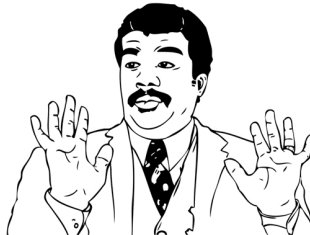
- Também pode ser sensível a bordas de subida, descida ou ambas no pino CCP;
- Quando o *timer* identifica o evento no pino CCP, ele armazena o valor da contagem nos registradores **GPTMTnR**;
- O valor permanece lá até a chegada de outro evento no pino;
- Conta até 24 bits (utilizando o *prescaler* como os 8 bits superiores);
- Detalhes de configuração (ver *Datasheet* seção 13.4.4 na página 973);

## Detalhes Adicionais

- Verifique o capítulo 13 no Datasheet do Tiva para outras configurações.

## Detalhes Adicionais

- Verifique o capítulo 13 no Datasheet do Tiva para outras configurações.



# Exemplos

## Exemplo

- Configurando o **Timer2** para contar **700ms**, apenas uma vez e ao final da contagem alternar o estado de um LED;
- Primeiro, verifique quantos bits vamos precisar e qual o valor do registrador de contagem e *prescale* (se necessário);
- Utilizar a expressão, para saber quantos ticks devemos contar:

$$\begin{aligned} \text{Tempo} &= (1/\text{sysclk}) * (\text{contagem} + 1) * (\text{prescale} + 1) \\ \text{Fazendo } (\text{contagem} + 1) * (\text{prescale} + 1) &= X \\ 700m &= (1/80M) * X \\ X &= 56.000.000 \end{aligned} \tag{1}$$

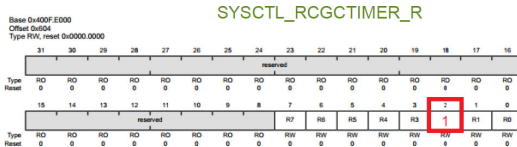
## Exemplo

- $X = 56.000.000$ 
  - 16 bits  $\rightarrow$  conta até  $2^{16} = 65536$ ;
  - 16 bits com *prescale* de 8 bits  $\rightarrow$  conta até  $2^{24} = 16.777.216$ ;
  - 32 bits  $\rightarrow$  conta até  $2^{32} = 4.294.967.296$ ;
- Se temos que contar até 56.000.000, então o único modo possível é contar 32 bits (onde não há *prescale*):
- Neste caso,  $X = (contagem + 1) = 56.000.000$   
 $contagem = 55.999.999$
- Esse é o valor que vamos colocar no registrador  
 $GPTMTAILR = 55.999.999$ , lembrando que como o *timer* opera em 32 bits, usa-se apenas o TimerA e o TimerB não pode ser utilizado.

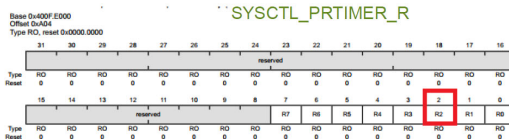


## Exemplo (seguindo os passos)

- 1 Como vamos utilizar o **timer2**, habilite o bit 2 no registrador **RCGCTIMER** e depois esperar enquanto bit 2 do registrador **PRTIMER** não está setado.



Setar este bit



Ficar testando  
este bit, até que  
seja 1

## Exemplo (seguindo os passos)

- Desabilite o **timer2** para fazer as configurações (adiante vamos habilitá-lo novamente);

GPTM Control (GPTMCTL)

16/32-bit Timer 0 base: 0x4003.0000

16/32-bit Timer 1 base: 0x4003.1000

16/32-bit Timer 2 base: 0x4003.2000

16/32-bit Timer 3 base: 0x4003.3000

16/32-bit Timer 4 base: 0x4003.4000

16/32-bit Timer 5 base: 0x4003.5000

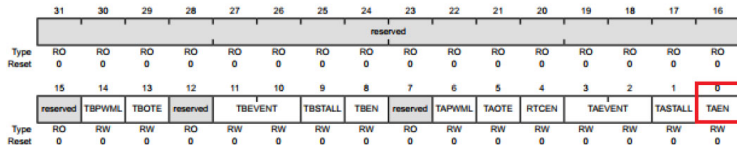
16/32-bit Timer 6 base: 0x400E.0000

16/32-bit Timer 7 base: 0x400E.1000

Offset 0x00C

Type RW, reset 0x0000.0000

TIMER2\_CTL\_R



Colocar 0 neste bit

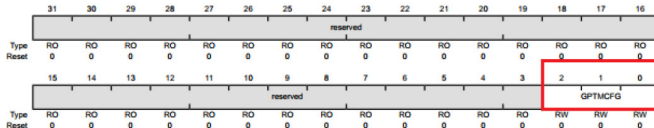
## Exemplo (seguindo os passos)

- 3 Conforme calculamos os valores, coloque o timer2 no modo **32 bits** então escrever **0x00** nos bits respectivos do registrador **GPTMCFG**.

### GPTM Configuration (GPTMCFG)

16/32-bit Timer 0 base: 0x4003.0000  
 16/32-bit Timer 1 base: 0x4003.1000  
 16/32-bit Timer 2 base: 0x4003.2000  
 16/32-bit Timer 3 base: 0x4003.3000  
 16/32-bit Timer 4 base: 0x4003.4000  
 16/32-bit Timer 5 base: 0x4003.5000  
 16/32-bit Timer 6 base: 0x400E.0000  
 16/32-bit Timer 7 base: 0x400E.1000  
 Offset 0x000  
 Type RW, reset 0x0000.0000

### TIMER2\_CFG\_R



Colocar 0x00  
nestes bits

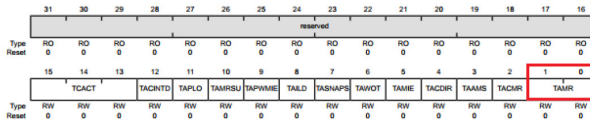
## Exemplo (seguindo os passos)

- 4 Como vamos usar o **timerA** e vamos contar apenas uma vez, vamos colocá-lo no **modo one-shot**, colocando 1 no campo respectivo do registrador **GPTMTAMR**.

GPTM Timer A Mode (GPTMTAMR)

16/32-bit Timer 0 base: 0x4003.0000  
16/32-bit Timer 1 base: 0x4003.1000  
16/32-bit Timer 2 base: 0x4003.2000  
16/32-bit Timer 3 base: 0x4003.3000  
16/32-bit Timer 4 base: 0x4003.4000  
16/32-bit Timer 5 base: 0x4003.5000  
16/32-bit Timer 6 base: 0x400E.0000  
16/32-bit Timer 7 base: 0x400E.1000  
Offset 0x004  
Type RW, Reset 0x0000.0000

TIMER2\_TAMR\_R



Colocar 0x01 nestes bits.

## Exemplo (seguindo os passos)

- 5 Carregue o valor de contagem no registrador **timerA** no registrador **GPTMTAILR**:

GPTM Timer A Interval Load (GPTMTAILR)

16/32-bit Timer 0 base: 0x4003.0000

16/32-bit Timer 1 base: 0x4003.1000

16/32-bit Timer 2 base: 0x4003.2000

16/32-bit Timer 3 base: 0x4003.3000

16/32-bit Timer 4 base: 0x4003.4000

16/32-bit Timer 5 base: 0x4003.5000

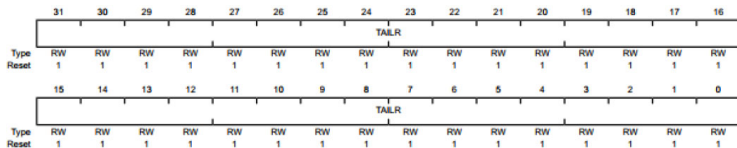
16/32-bit Timer 6 base: 0x400E.0000

16/32-bit Timer 7 base: 0x400E.1000

Offset 0x028

Type RW, reset 0xFFFF.FFFF

TIMER2\_TAILR\_R



Colocar 55.999.999  
neste registrador.

## Exemplo (seguindo os passos)

- 6 Como não temos *prescale*, deixe o registrador **GPTMTAPR** zerado.

GPTM Timer A Prescale (GPTMTAPR)

16/32-bit Timer 0 base: 0x4003.0000

16/32-bit Timer 1 base: 0x4003.1000

16/32-bit Timer 2 base: 0x4003.2000

16/32-bit Timer 3 base: 0x4003.3000

16/32-bit Timer 4 base: 0x4003.4000

16/32-bit Timer 5 base: 0x4003.5000

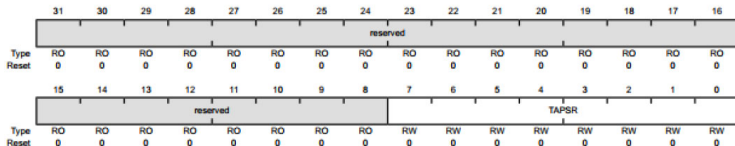
16/32-bit Timer 6 base: 0x400E.0000

16/32-bit Timer 7 base: 0x400E.1000

Offset 0x038

Type RW, reset 0x0000.0000

TIMER2\_TAPR\_R



Colocar zero neste  
registrador.

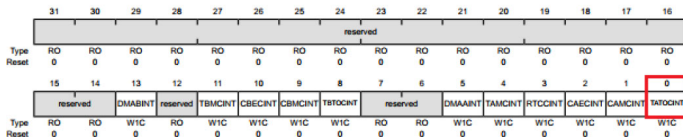
## Exemplo (seguindo os passos)

- 7 Como vamos utilizar o timerA, sete o bit **TnTOCINT** no registrador **GPTMICR**, para garantir que a primeira interrupção seja atendida.

GPTM Interrupt Clear (GPTMICR)

16/32-bit Timer 0 base: 0x4003.0000  
16/32-bit Timer 1 base: 0x4003.1000  
**16/32-bit Timer 2 base: 0x4003.2000**  
16/32-bit Timer 3 base: 0x4003.3000  
16/32-bit Timer 4 base: 0x4003.4000  
16/32-bit Timer 5 base: 0x4003.5000  
16/32-bit Timer 6 base: 0x400E.0000  
16/32-bit Timer 7 base: 0x400E.1000  
**Offset 0x024**  
type W1C, reset 0x0000.0000

TIMER2\_ICR\_R



Setar este bit para 1 para zerar o flag de atendimento de interrupção (ACK)

## Exemplo (seguindo os passos)

- 8 a) Como vamos utilizar interrupção para estouro do timer, escreva 1 no bit **TATOIM** do registrador **GPTMIMR**:

GPTM Interrupt Mask (GPTMIMR)

16/32-bit Timer 0 base: 0x4003.0000

16/32-bit Timer 1 base: 0x4003.1000

16/32-bit Timer 2 base: 0x4003.2000

16/32-bit Timer 3 base: 0x4003.3000

16/32-bit Timer 4 base: 0x4003.4000

16/32-bit Timer 5 base: 0x4003.5000

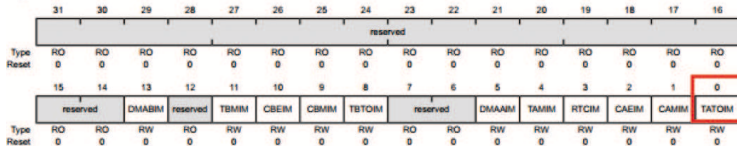
16/32-bit Timer 6 base: 0x400E.0000

16/32-bit Timer 7 base: 0x400E.1000

Offset 0x018

type RW, reset 0x0000.0000

TIMER2\_IMR\_R



Setar este bit para 1 para setar a interrupção do TimerA



## Exemplo (seguindo os passos)

- 8 b) Sete a **prioridade** da interrupção do *timer* respectivo no respectivo registrador *NVIC Priority Register*:
- Primeiramente, verifique qual é o número da interrupção do Timer2A na tabela 2-9 do *datasheet*.

Table 2-9. Interrupts (continued)

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Vector Address or Offset	Description
38	22	0x0000.0098	16/32-Bit Timer 1B
39	23	0x0000.009C	16/32-Bit Timer 2A
40	24	0x0000.00A0	16/32-Bit Timer 2B
41	25	0x0000.00A4	Analog Comparators 0
42	26	0x0000.00A8	Analog Comparators 1
43	27	0x0000.00AC	Analog Comparators 2
44	28	0x0000.00B0	System Control
45	29	0x0000.00B4	Flash Memory Control
46	30	0x0000.00B8	GPIO Port F
47	31	0x0000.00BC	GPIO Port G
48	32	0x0000.00C0	GPIO Port H
49	33	0x0000.00C4	UART2
50	34	0x0000.00C8	SSI1
51	35	0x0000.00CC	16/32-Bit Timer 3A
52	36	0x0000.00D0	16/32-Bit Timer 3B

Interrupção número 23, vamos utilizar este número para 8b e 8c.

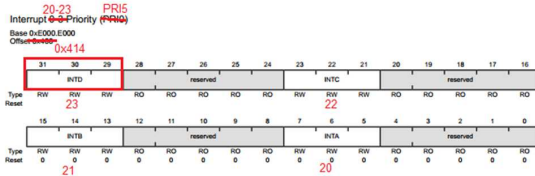
## Exemplo (seguindo os passos)

- 8 b) Sete a **prioridade** da interrupção do *timer* respectivo no respectivo registrador *NVIC Priority Register*:
- Encontre qual o registrador de prioridade do NVIC tem a prioridade para a interrupção de nº23 (timer2A)

Register 24: Interrupt 0-3 Priority (PRI0), offset 0x400  
Register 25: Interrupt 4-7 Priority (PRI1), offset 0x404  
Register 26: Interrupt 8-11 Priority (PRI2), offset 0x408  
Register 27: Interrupt 12-15 Priority (PRI3), offset 0x40C  
Register 28: Interrupt 16-19 Priority (PRI4), offset 0x410  
**Register 29: Interrupt 20-23 Priority (PRI5), offset 0x414**  
Register 30: Interrupt 24-27 Priority (PRI6), offset 0x418  
Register 31: Interrupt 28-31 Priority (PRI7), offset 0x41C  
Register 32: Interrupt 32-35 Priority (PRI8), offset 0x420  
Register 33: Interrupt 36-39 Priority (PRI9), offset 0x424  
Register 34: Interrupt 40-43 Priority (PRI10), offset 0x428  
Register 35: Interrupt 44-47 Priority (PRI11), offset 0x42C  
Register 36: Interrupt 48-51 Priority (PRI12), offset 0x430  
Register 37: Interrupt 52-55 Priority (PRI13), offset 0x434  
Register 38: Interrupt 56-59 Priority (PRI14), offset 0x438  
Register 39: Interrupt 60-63 Priority (PRI15), offset 0x43C

## Exemplo (seguindo os passos)

- 8 b) Sete a **prioridade** da interrupção do *timer* respectivo no respectivo registrador *NVIC Priority Register*:
- Coloque o valor da prioridade no campo da respectiva interrupção (neste caso vamos colocar prioridade 4)



Fazer um deslocamento do valor 4 em 29 bits para facilitar a escrita no respectivo campo.

```
MOV R0, #4
```

```
LSL R0, R0, #29
```

## Exemplo (seguindo os passos)

- 8 c) **Habilite** a interrupção do timer respectivo no respectivo registrador *NVIC Interrupt Enable Register*:
- Encontre qual o registrador de habilitação do NVIC habilita a interrupção de nº23 (timer2A)

Register 4: Interrupt 0-31 Set Enable (EN0), offset 0x100

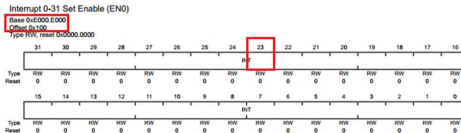
Register 5: Interrupt 32-63 Set Enable (EN1), offset 0x104

Register 6: Interrupt 64-95 Set Enable (EN2), offset 0x108

Register 7: Interrupt 96-113 Set Enable (EN3), offset 0x10C

## Exemplo (seguindo os passos)

- 8 c) **Habilite** a interrupção do *timer* respectivo no respectivo registrador *NVIC Interrupt Enable Register*:
- Sete o bit no campo da respectiva interrupção (neste caso vamos colocar prioridade 4)



Fazer um deslocamento do valor 1 em 23 bits  
para facilitar a escrita no respectivo campo.

```
MOV R0, #1
LSL R0, R0, #23
```

## Exemplo (seguindo os passos)

- 9 Habilite o bit **TAEN** no registrador **GPTMCTL** para começar o *timer*. A partir deste momento o timer fará a contagem e quando estourar cairá na rotina de tratamento de interrupção.

GPTM Control (GPTMCTL)

16/32-bit Timer 0 base: 0x4003.0000

16/32-bit Timer 1 base: 0x4003.1000

16/32-bit Timer 2 base: 0x4003.2000

16/32-bit Timer 3 base: 0x4003.3000

16/32-bit Timer 4 base: 0x4003.4000

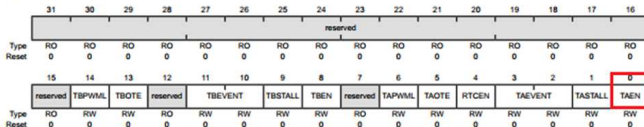
16/32-bit Timer 5 base: 0x4003.5000

16/32-bit Timer 6 base: 0x400E.0000

16/32-bit Timer 7 base: 0x400E.1000

Offset 0x000

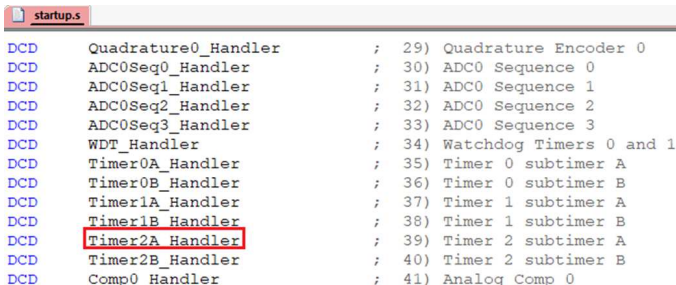
Type RW, reset 0x0000.0000



Colocar 1 neste bit para habilitar a contagem

## Exemplo (seguindo os passos)

- 10 Crie a rotina de tratamento de interrupção para o **timer2A**.  
Procurar no arquivo **startup.s** qual é o nome da ISR para o estouro do **timer2A**.



```
startup.s
DCD    Quadrature0_Handler      ; 29) Quadrature Encoder 0
DCD    ADC0Seq0_Handler        ; 30) ADC0 Sequence 0
DCD    ADC0Seq1_Handler        ; 31) ADC0 Sequence 1
DCD    ADC0Seq2_Handler        ; 32) ADC0 Sequence 2
DCD    ADC0Seq3_Handler        ; 33) ADC0 Sequence 3
DCD    WDT_Handler             ; 34) Watchdog Timers 0 and 1
DCD    Timer0A_Handler          ; 35) Timer 0 subtimer A
DCD    Timer0B_Handler          ; 36) Timer 0 subtimer B
DCD    Timer1A_Handler          ; 37) Timer 1 subtimer A
DCD    Timer1B_Handler          ; 38) Timer 1 subtimer B
DCD    Timer2A_Handler        ; 39) Timer 2 subtimer A
DCD    Timer2B_Handler          ; 40) Timer 2 subtimer B
DCD    Comp0_Handler           ; 41) Analog Comp 0
```

Definir esta função em qualquer arquivo  
(preferencialmente não no startup.s) fazer o  
**EXPORT** no cabeçalho do mesmo.

## Exemplo (seguindo os passos)

- 10 Crie a rotina de tratamento de interrupção para o **timer2A**.  
Procurar no arquivo **startup.s** qual é o nome da ISR para o estouro do **timer2A**.
- Primeiramente, dentro da ISR devemos realizar o ACK escrevendo 1 no bit **TATOCINT**, que limpa o *flag* de interrupção no registrador **GPTMICR** (e limpa o bit **TATORIS** do registrador **GPTMRIS**), de maneira que novas interrupções possam ser tratadas. Depois, devemos inverter o estado do LED.

```
Timer2A_Handler
LDR R1, =TIMER2_ICR_R
MOV R0, #1
STR R0, [R1]
PUSH {LR}
BL PortN_Invertepino0 ; Chama a rotina que faz um
toggle no led
POP {LR} ; retorno da interrupção
BX LR
```



## Exemplo (seguindo os passos)

- Como este timer está configurado para o modo *one shot*, o timer só contará uma vez e não contará novamente até que armemos novamente;
- Assim, os passos 1 a 8 podem ser feitos em uma função de inicialização;
- Já o passo 9 pode ser feito para armar o timer a cada vez que um botão seja pressionado;
- Por exemplo, coloque na rotina de tratamento de interrupção de quando um botão foi pressionado, ou em um laço que verifica se o botão foi pressionado.

# Dúvidas?