UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Departamento Acadêmico de Elétrica – DAELE
Cursos Superiores de Engenharia Elétrica e Engenharia de Computação

PARTE III

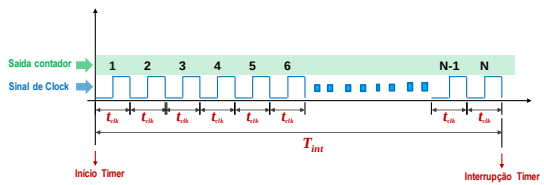
Timer A – Básico

Fonte:
MSP430x2xx Family Users Guide

Prof. Dr. Fábio L. Bertotti
bertotti@utfpr.edu.br

INTRODUÇÃO

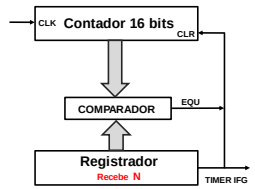
- **TEMPORIZADORES**
  - Princípio Geral de funcionamento

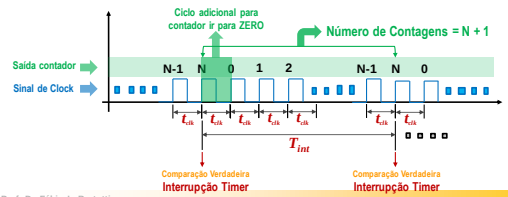


$$T_{int} = \sum_{n=0}^{n=N} t_{clk}$$

INTRODUÇÃO

- **TEMPORIZADORES**
  - **MODO COMPARAÇÃO**
    - DETALHE Nº DE CONTAGENS





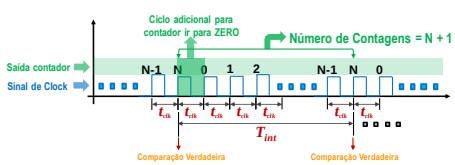
INTRODUÇÃO

- **TEMPORIZADORES**
  - **MODO COMPARAÇÃO**
    - **CÁLCULO de TEMPO para INTERRUPÇÃO:**

$$VAL_{REGISTRADOR} = N = (tempo.Freq\_Clock) - 1$$

(Número de Contagens)

$$tempo = \frac{VAL_{REGISTRADOR} + 1}{Freq\_Clock}$$

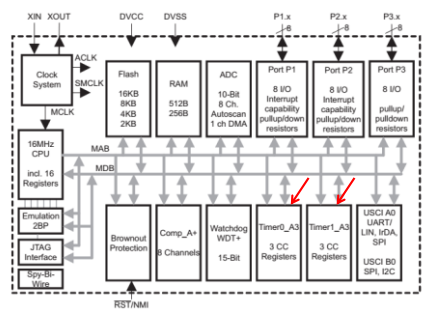


Timer A

- Características
  - Temporizador/Contador de 16 bits
  - **4 Modos de operação:**
    - Contador
    - Captura
    - **Comparação**
    - PWM
  - **3 registradores** de Captura/Comparação (MSP430G2553)

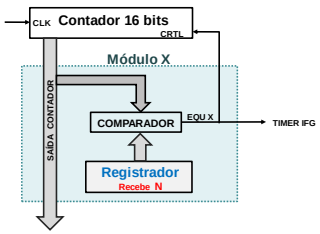
Timer A

- MSP430G2553

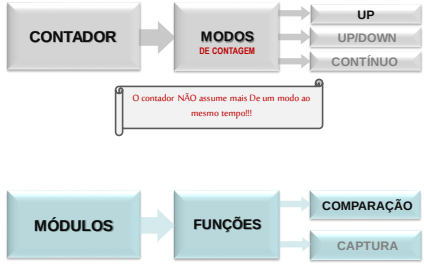


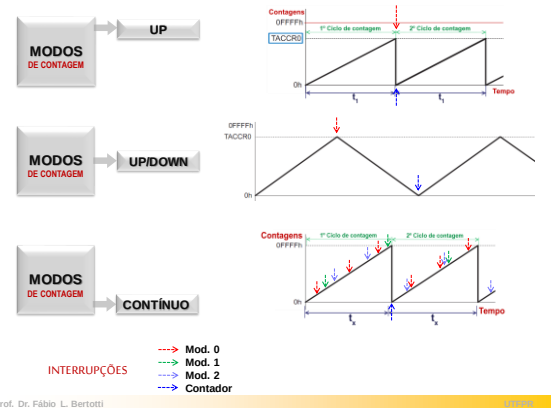
Timer A

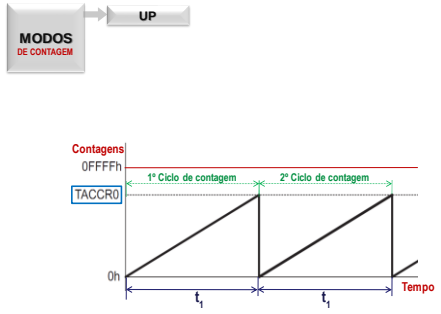
- MÓDULOS
  - Configurados na **FUNÇÃO COMPARAÇÃO** ou **CAPTURA**
  - **Exemplo:** Módulo X **configurado na função COMPARAÇÃO:**

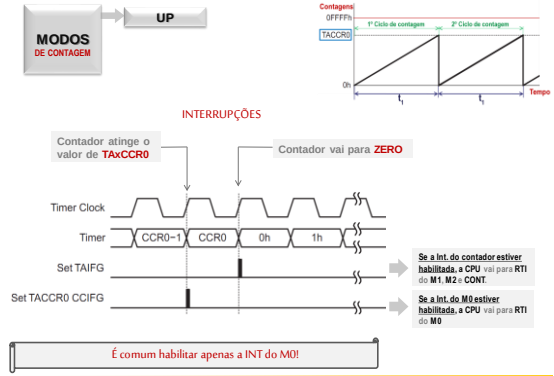


Timer A: Contador e Módulos

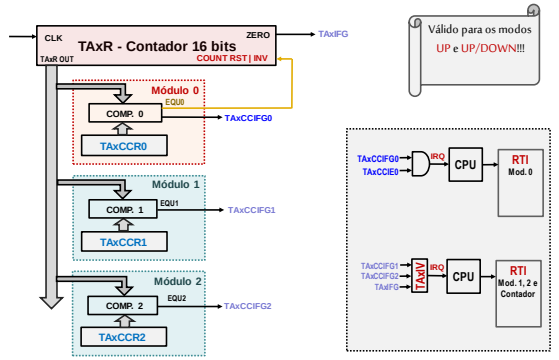








Timer A - Diagrama em blocos simplificado - COMPARAÇÃO



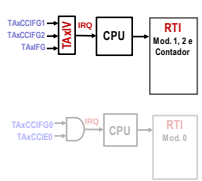
Timer A

- INTERRUPÇÕES Módulos 1, 2 e Contador
  - TAxIV (Timer_A Interrupt Vector Register)

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
r0	r0	r0	r0	r0	r0	r0	r0

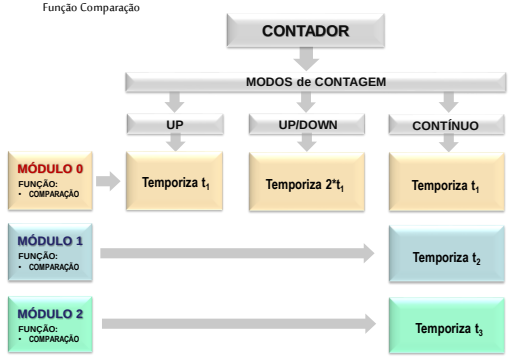
7	6	5	4	3	2	1	0
0	0	0	0	TAIVx			0
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r0

TAIV Contents	Interrupt Source	Interrupt Flag	Interrupt Priority
00h	No interrupt pending	-	
02h	Capture/compare 1	TACCR1 CCIFG	Highest
04h	Capture/compare 2**	TACCR2 CCIFG	
06h	Reserved	-	
08h	Reserved	-	
0Ah	Timer overflow	TAIFG	
0Ch	Reserved	-	
0Eh	Reserved	-	Lowest



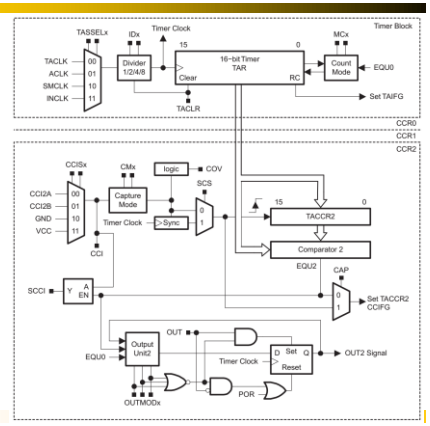
Timer A:

Função Comparação



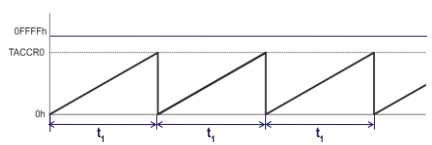
Timer A

- **Diagrama em blocos COMPLETO**



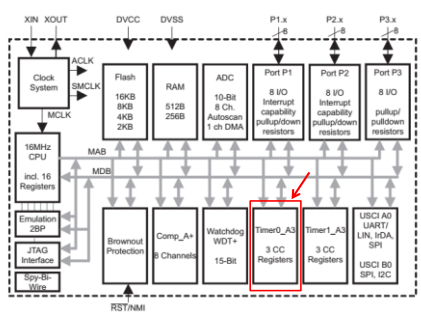
Timer A

- EXEMPLO 1: Modo UP
  - **Timer A** configurado para gerar interrupções a cada **1s** e na **RTI** mudar o estado do LED Vermelho (P1.0), usando como fonte de clock o sinal **ACLK**, proveniente de **LFXT1** (cristal de 32.768 Hz).



Timer A

- MSP430G2553 – Timer0



Exemplo 1 – Timer A

- 1º Passo: Criação de função para as configurações iniciais.

```
void config_ini(void){
  WDTCTL = WDTPW | WDTHOLD; // Para o contador do watchdog timer

  // Configuracoes do BCS
  // MCLK = DCOCLK - 16 MHz
  // ACLK = LFXT1CLK = 32768 Hz
  // SMCLK = DCOCLK / 8 - 2 MHz
  DCOCTL = CALDCO_16MHZ;    // Freq. Calibrada de 16 MHz
  BCSCTL1 = CALBC1_16MHZ;
  BCSCTL2 = DIVS0 + DIVS1;  // Fator divisao = 8 para SMCLK
  BCSCTL3 = XCAP0 + XCAP1;  // Capacitor do cristal -12.5 pF

  while(BCSCTL3 & LFXT1OF);

  __enable_interrupt();  // seta o bit GIE - permite geracao de interrupcoes
}
```

Exemplo 1 - Timer A

- **2º Passo:** Criação de função para as configurações das Portas de I/O.

```
void ini_P1_P2(void){
      // Todos os pinos da Porta 1 como saida
      P1DIR = BIT0 + BIT1 + BIT2 + BIT3 + BIT4 + BIT5 + BIT6 + BIT7;

      P1OUT = 0; // Todas as saidas da Porta 1 em nivel baixo
                 // LED vermelho apagado

      // Todos os pinos da Porta 2 como saida, exceto P2.6 e P2.7 (LFXT1)

      P2DIR = 0xFF;  // P2: Todos os bits como saida em nivel baixo
      P2OUT = 0;
}
```


Exemplo 1 - Timer A

- **3º Passo:** Encontrar valor para TA0CCR0

$$\textit{Base de tempo} \qquad t_b = \frac{FDIV}{ACLK}$$

Fator de Divisão

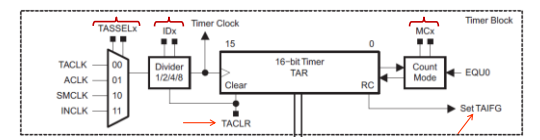
Fonte de Clock

$$TACCR0 = \frac{tempo}{t_b} = \frac{tempo.ACLK}{FDIV} - 1$$

Exemplo 1 - Timer A

- **4º Passo:** Configurar TA0CTL

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)



Exemplo 1 - Timer A

- **4º Passo:** Configurar TA0CTL

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused	Bits 15-10	Unused
TASSELx	Bits 9-8	Timer_A clock source select
		00 TACLK
	→	01 ACLK
		10 SMCLK
		11 INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)
IDx	Bits 7-6	Input divider. These bits select the divider for the input clock.
	→	00 /1
		01 /2
		10 /4
		11 /8

```
TA0CTL = TASSEL0 +
```


Exemplo 1 - Timer A

- **4º Passo:** Configurar TA0CTL

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

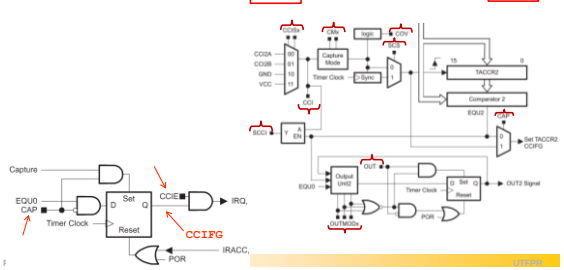
MCx	Bits 5-4	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
		00 Stop mode: the timer is halted.
	→	01 Up mode: the timer counts up to TACCR0.
		10 Continuous mode: the timer counts up to 0FFFFh.
		11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.
Unused	Bit 3	Unused
TACLR	Bit 2	Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.
TAIE	Bit 1	Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
	→	0 Interrupt disabled
		1 Interrupt enabled
TAIFG	Bit 0	Timer_A interrupt flag
	→	0 No interrupt pending
		1 Interrupt pending

```
TA0CTL = TASSEL0 + MC0;
```


Exemplo 1 - Timer A

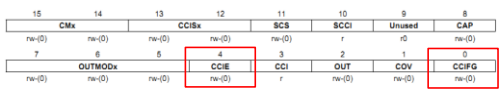
- **5º Passo:** Configurar TA0CCTL0

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)



Exemplo 1 - Timer A

- **5º Passo:** Configurar TA0CCTL0



15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCIE Bit 4 Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
→ 0 Interrupt disabled
→ 1 Interrupt enabled

CCIFG Bit 0 Capture/compare interrupt flag
→ 0 No interrupt pending
1 Interrupt pending

```
TA0CCTL0 = CCIE;
```


Exemplo 1 - Timer A

- Configurações iniciais do Timer0 A

```
void ini_TA0(void){

   // TAclk = ACLK = 32768 Hz, Modo UP, Interrup. a cada 1s

   TA0CTL = TASSEL0 + MC0;

   TA0CCTL0 = CCIE;

   TA0CCR0 = 32767;

}
```


Exemplo 1 - Timer A

- **6º Passo:** RTI do Timer0 A

```
#pragma vector=TIMER0_A0_VECTOR
__interrupt void TA0CCR0_RTI(void){

      P1OUT ^= BIT0;      // Alterna LED VM

}
```


Exemplo 1 - Timer A

- **Início do código**

```
#include <msp430.h>

void config_ini(void);   // Prototipos das funcoes
void ini_P1_P2(void);
void ini_TA0(void);


void main(void){
    config_ini();
    ini_P1_P2();
    ini_TA0();

    do{
        // Não faz nada!
    }while(1);
}
```


Timer A

- **Exercício 1**

  - **Implemente um relógio usando o Timer0 A.**
    - Use o sinal ACLK como fonte de clock.
    - Na RTI do Módulo 0 atualize as variáveis de tempo SEGUNDO, MINUTO e HORA.
    - Monitore as variáveis no Code Composer.

UTFPR Tecnológica há mais de 100
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Departamento Acadêmico de Elétrica – DAELE
Cursos Superiores de Engenharia Elétrica e Engenharia de Computação

Processamento de chaves

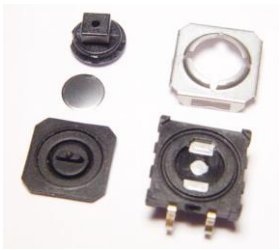
Técnica Debouncer

Fonte:
MSP430x2xx Family Users Guide

Prof. Dr. Fábio L. Bertotti
bertotti@utfpr.edu.br

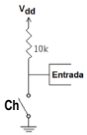
Processamento de Chave

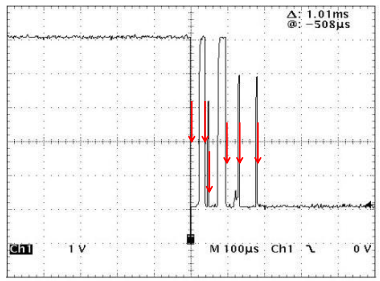
- Chave Eletromecânica



Processamento de Chave

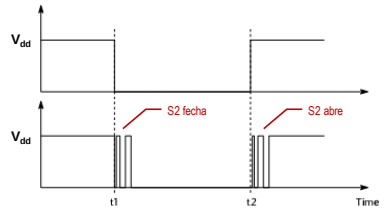
- Efeito BOUNCE





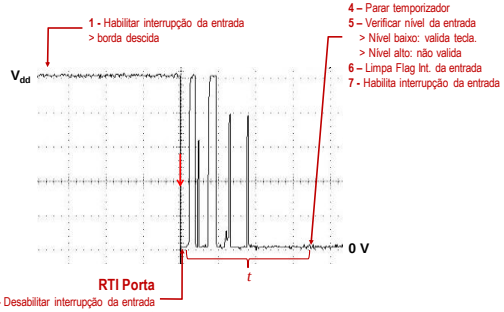
Processamento de Chave

- Efeito na saída do schimidt trigger da Porta



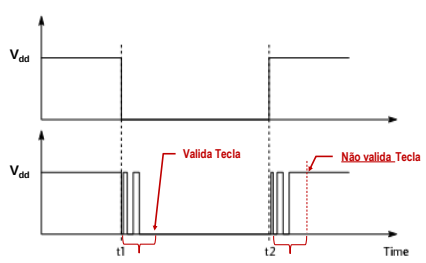
Processamento de Chave

- **Debounce:** Estratégia



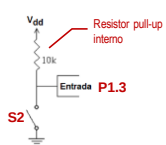
Processamento de Chave

- **Debounce:** ANÁLISE



EXEMPLO 2 – Debouncer

- Debouncer: Implementação de debouncer para a chave S2
  - Cada vez que S2 for pressionada, o estado do led vermelho deve ser alterado.
  - Usar **Timer0 A** para temporizar t (5 ms)
    - Clock: **SMCLK** ~ 2 MHz
  - RTI da Porta 1 ?
  - RTI do M0 do Timer0 A ?
  - Desenvolvimento de código no Code Composer....



EXEMPLO 2 - Debouncer

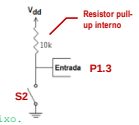
- **1º Passo:** Criação de função para as config. iniciais...

```
void ini_ucon(void){
  WDTCTL = WDTPW | WDTHOLD;

  // Configuracoes do BCS
  // MCLK = DCOCLK ~ 16 MHz
  // ACLK = LFXT1CLK = 32768 Hz
  // SMCLK = DCOCLK / 8 ~ 2 MHz
  DCOCTL = CALDCO_16MHZ;
  BCSCTL1 = CALBC1_16MHZ;
  BCSCTL2 = DIVS0 + DIVS1;
  BCSCTL3 = XCAP0 + XCAP1;

  while(BCSCTL3 & LFXT1OF);
  __enable_interrupt();
}
```


EXEMPLO 2 - Debouncer

- **2º Passo:** Criação de função para as configurações das Portas de I/O.

```
void ini_P1_P2(void){
  // BIT3 como entrada e os demais como saida
  P1DIR = BIT0 + BIT1 + BIT2 + BIT4 + BIT5 + BIT6 + BIT7;
  P1REN = BIT3;  // Resistor do BIT3 habilitado.
  P1OUT = BIT3;  // Resistor de pull-up para BIT3,
                 // os demais como saida em nivel logico baixo.
  P1IES = BIT3;  // Interrupcao por borda de descida.
  P1IFG = 0;     // Limpa as flags da P1, evitando que uma
                 // interrupcao ocorra de forma indevida.
  P1IE = BIT3;   // Interrupcao do BIT3 da P1 habilitada.
  P2DIR = 0xFF;  // P2: Todos os bits como saida em nivel baixo.
  P2OUT = 0x00;  // Pinos 18 e 19 com função nativa (XIN/XOUT).
}
```



EXEMPLO 2 - Debouncer

- **3º Passo:** Inicialização do Timer0 A

```
void ini_Timer0(void){
    /* Configuracoes iniciais do Timer0 para o Debouncer de S2
    *
    * CONTADOR
    *      - Fonte de Clock: SMCLK ~ 2 MHz
    *      - Fdiv clock: 1
    *      - Modo cont.: Inicialmente >>>> PARADO!
    *      - Int. cont.: desabilitada
    *
    * MODULO 0
    *      - Modo: Comparacao (default)
    *      - Interrupcao: Habilitada
    *      - TA0CCR0 = (SMCLK * 0.005) - 1 = 9999
    *
    */
      TA0CTL = TASSEL1;
      TA0CCTL0 = CCIE;
      TA0CCR0 = 9999;
}
```


EXEMPLO 2 - Debouncer

- **4º Passo:** RTI da Porta 1

```
// RTI da PORTA 1
#pragma  vector=PORT1_VECTOR
__interrupt void P1_RTI(void){

      P1IFG &= ~BIT3; // Opcional aqui: Limpa flag de int.
                      // Deve ser feito no final do processo de debounce.
      P1IE &= ~BIT3; // Desabilita int. do BIT3 da P1

      // Inicia Timer0 para t ~ 5ms
      TA0CTL |= MC0;
}
```


EXEMPLO 2 - Debouncer

- **5º Passo:** RTI do M0 do Timer0 A

```
// RTI do M0 do Timer0
#pragma  vector=TIMER0_A0_VECTOR
__interrupt void RTI_do_M0_do_Timer0(void){
      // Flag de int. é limpa automaticamente
      TA0CTL &= ~MC0;  // Para o Timer0 -> vai para modo STOP

      if( (~P1IN) & BIT3 ){  // Verifica se tecla realmente foi pressionada
          P1OUT ^= BIT0;     // Tecla press. -> alterna estado do Led VM
      }

      P1IFG &= ~BIT3;  // Obrigatorio limpar flag aqui!

      P1IE |= BIT3;    // Habilita int. do BIT3 da P1
}
```

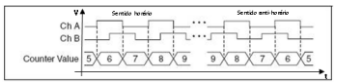

EXEMPLO 2 - Debouncer

- **6º Passo:** Programa Principal

```
#include <msp430.h>

void ini_ucon(void);
void ini_P1_P2(void);
void ini_Timer0(void);

void main(void){

    ini_ucon();
    ini_P1_P2();
    ini_Timer0();

    do{
        // Loop infinito
    }while(1);

}
```

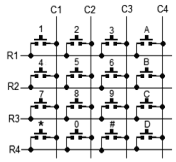
Timer A

- Efetue a Leitura de um Encoder
  - O encoder provê os seguintes sinais A (Ch A) e B (Ch B) ao microcontrolador, cujos sinais são conectados em 2 pinos da Porta 1 (P1.6 e P1.7) do microcontrolador MSP430G2553. A cada passo do encoder no sentido horário/anti-horário uma variável deve ser incrementada/decrementada de uma unidade.



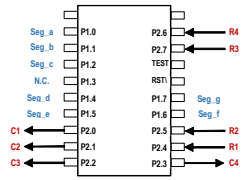
Leitura de Teclado 4x4

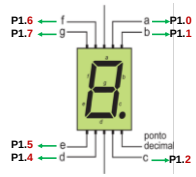
- Dicas para Leitura de Teclado tipo Matriz 4x4 com Debouncer



Teclado Matricial

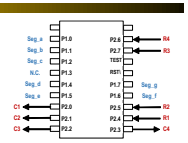
- Conexão do teclado e display de 7 segmentos

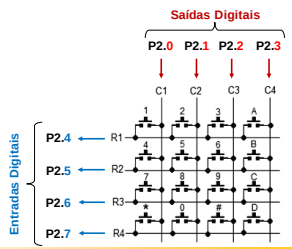




Teclado Matricial

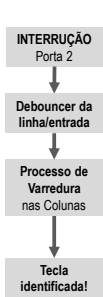
- Configurações Iniciais das Portas
  - Opção 1:
    - P2.4-P2.7 - entradas com resistor de PULL-UP
      - Interrupção por borda de DESCIDA.
    - P2.0-P2.3 – Saídas em nível BAIXO
  - Opção 2:
    - P2.4-P2.7 - entradas com resistor de PULL-DOWN
      - Interrupção por borda de SUBIDA
    - P2.0-P2.3 – Saídas em nível ALTO

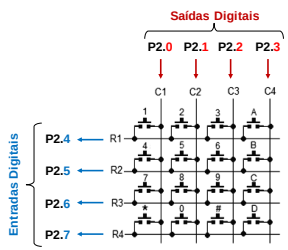




Teclado Matricial

- Processo de Leitura





UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Departamento Acadêmico de Elétrica – DAELE
Cursos Superiores de Engenharia Elétrica e Engenharia de Computação

TIMER A

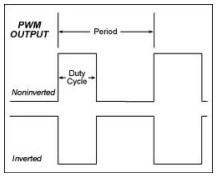
Geração de Sinais PWM

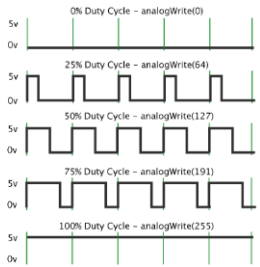
Fonte:
MSP430x2xx Family Users Guide

Prof. Dr. Fábio L. Bertotti
bertotti@utfpr.edu.br

Timer A
PWM

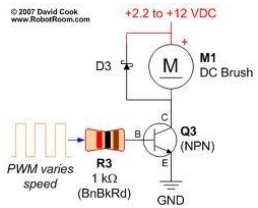
- **Conceito**





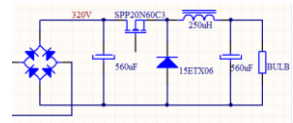
Timer A
PWM

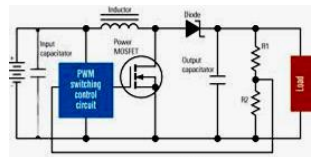
- **Aplicações**
  - **Controle de velocidade de motores DC**



Timer A
PWM

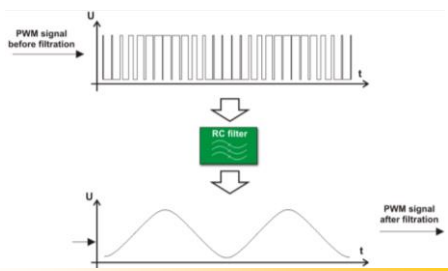
- **Aplicações**
  - **Conversores DC-DC**



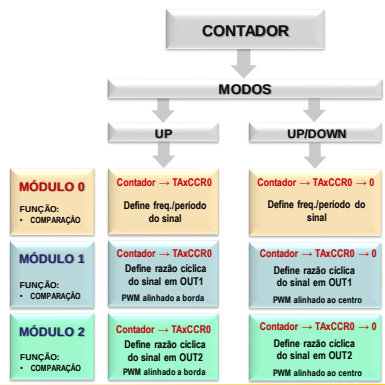


Timer A
PWM

- **Aplicações**
  - **Geração de sinais**

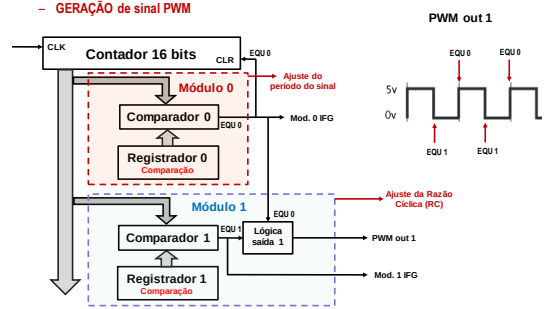


Timer A: MODO PWM

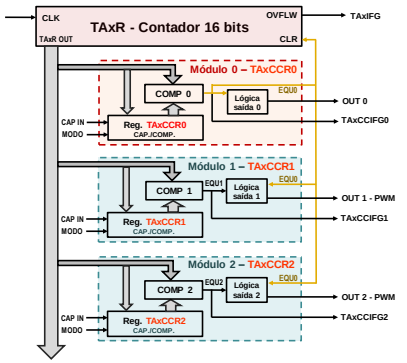


INTRODUÇÃO

- **TEMPORIZADORES**
  - **GERAÇÃO de sinal PWM**

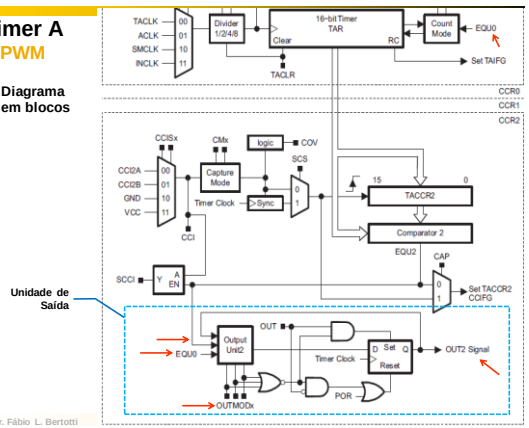


Timer A - Diagrama em blocos resumido



Timer A
PWM

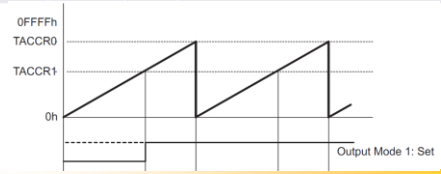
- Diagrama em blocos



Timer A
PWM

- Modos de Saída

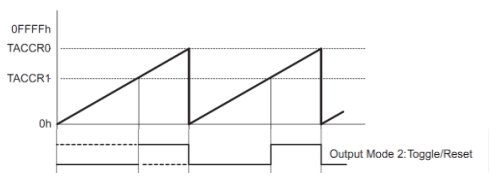
Modo	OUTMODX	Func.	Descrição
0	000	Saída Direta	OUTx é definido pelo estado do bit OUT de TACCTLx
1	001	Setar	OUTx é setado quando TAR = TACCRx OBS: A saída permanece setada até que outro modo seja selecionado



Timer A
PWM

- Modos de Saída

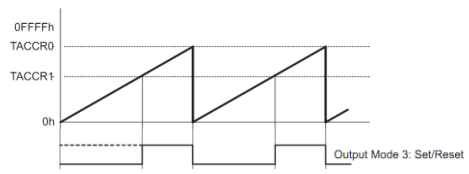
Modo	OUTMODX	Func.	Descrição
2	010	Inverter/ Resetar	OUTx é invertido quando TAR = TACCR1 e é resetado quando TAR = TACCR0



Timer A
PWM

- Modos de Saída

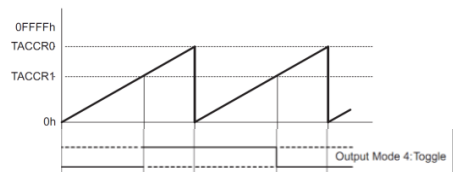
Modo	OUTMODX	Func.	Descrição
3	011	Setar/ Resetar	OUTx é Setado quando TAR = TACCR1 e é Resetado quando TAR = TACCR0



Timer A
PWM

- Modos de Saída

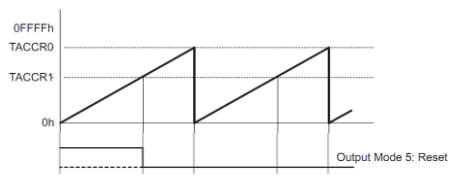
Modo	OUTMODX	Func.	Descrição
4	100	Inverter	OUTx é invertido quando TAR = TACCR1 OBS: o período do sinal é o dobro daquele definido nos modos 2 e 3



Timer A
PWM

- **Modos de Saída**

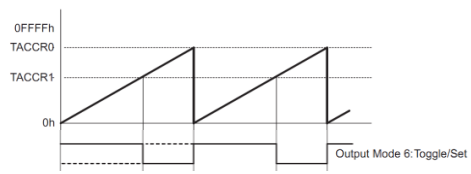
Modo	OUTMODX	Func.	Descrição
5	101	Resetar	OUTx é Resetada quando TAR = TACCR1 OBS: OUTx permanece resetado até que outro modo seja selecionado



Timer A
PWM

- **Modos de Saída**

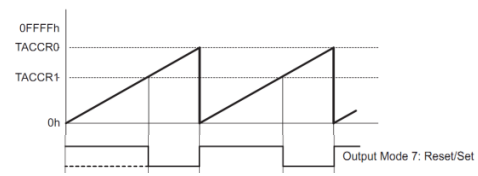
Modo	OUTMODX	Func.	Descrição
6	110	Inverter/ Setar	OUTx é Invertido quando TAR = TACCR1 e Setado quando TAR = TACCR0



Timer A
PWM

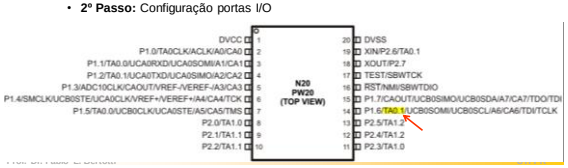
- **Modos de Saída**

Modo	OUTMODX	Func.	Descrição
7	111	Resetar/ Setar	OUTx é Resetado quando TAR = TACCR1 e Setado quando TAR = TACCR0



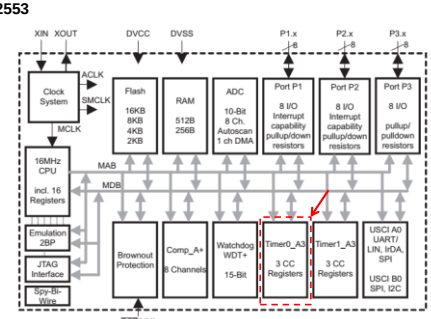
Timer A
PWM

- EXEMPLO 3
  - O **Timer0 A** do microcontrolador MSP430G2553 deve ser configurado no Modo PWM para controlar a luminosidade do LED Verde. Um sinal PWM com frequência de **100 Hz** e com largura de pulso ajustável e 0 a 100% deve ser gerado. Cada vez que **S2** for pressionada a razão cíclica deve ser aumentada de 25%, partindo de 0 a 100%.
    - **1º Passo:** Idem ao exemplo 1 e 2
    - **2º Passo:** Configuração portas I/O



Timer A: PWM
EXEMPLO 3

- **MSP430G2553**



Timer A: PWM
EXEMPLO 3

- **P1.6 do MSP430G2553**

Table 19. Port P1 (P1.5 to P1.7) Pin Functions

PIN NAME (P1.x)	x	FUNCTION	CONTROL BITS AND SIGNALS[1]					
			P1DIR.x	P1SEL.x	P1SEL2.x	ADC10AE.x INCH.x=1[2]	JTAG Mode	CAPD.y
P1.6/	6	P1.x (I/O)	I: 0; O: 1	0	0	0	0	0
TA0.1/		TA0.1	1	1	0	0	0	0
UCB0SOMI/		UCB0SOMI	from USCI	1	1	0	0	0
UCB0SCL/		UCB0SCL	from USCI	1	1	0	0	0
A6[2]/		A6	X	X	X	1 (y = 6)	0	0
CA6		CA6	X	X	X	0	0	1 (y = 6)
TDI/TCLK/		TDI/TCLK	X	X	X	0	1	0
Pin Osc		Capacitive sensing	X	0	1	0	0	0

Timer A: PWM

EXEMPLO 3

- **2º Passo:** Configuração portas I/O

```
void ini_P1_P2(void){
      // BIT3 como entrada e os demais como saida
    P1DIR = BIT0 + BIT1 + BIT2 + BIT4 + BIT5 + BIT6 + BIT7;
    P1REN = BIT3; // Resistor do BIT3 de P1 habilitado
    P1OUT = BIT3; // Resistor de pull-up para BIT3, os demais como saida
                  // em nivel logico baixo
    P1SEL |= BIT6; // saida de sinal PWM no pino associado a TA0.1
    P1IES = BIT3;  // Interrupcao por borda de descida
    P1IFG = 0; // limpa as flags da P1, evitando que uma interrupcao
               // ocorra de forma indevida
    P1IE = BIT3;   // Interrupcao do BIT3 da P1 habilitada
    P2DIR = 0xFF; // P2: Todos os bits como saida em nivel baixo, exceto os
    P2OUT = 0x00; // pinos 18 e 19, onde mentem-se as funções XIN e XOUT
  }
```


Timer A: PWM

EXEMPLO 3

- **3º Passo: Configurar TA0CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused Bits 15-10 Unused

TASSELx Bits 9-8 Timer_A clock source select
- 00 TACLK
- → 01 ACLK
- 10 SMCLK
- 11 INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)

IDx Bits 7-6 Input divider. These bits select the divider for the input clock.
- → 00 /1
- 01 /2
- 10 /4
- 11 /8

```
TA0CTL = TASSEL0 +
```


Timer A: PWM

EXEMPLO 3

- **3º Passo: Configurar TA0CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

MCx Bits 5-4 Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
- 00 Stop mode: the timer is halted.
- → 01 Up mode: the timer counts up to TACCR0.
- 10 Continuous mode: the timer counts up to 0FFFFh.
- 11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.

Unused Bit 3 Unused

TACLR Bit 2 Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.

TAIE Bit 1 Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
- → 0 Interrupt disabled
- 1 Interrupt enabled

TAIFG Bit 0 Timer_A interrupt flag
- → 0 No interrupt pending
- 1 Interrupt pending

```
TA0CTL = TASSEL0 + MC0;
```


Timer A: PWM

EXEMPLO 3

- **4º Passo: Configurar TA0CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx Bit 15-14 Capture mode
- → 00 No capture
- 01 Capture on rising edge
- 10 Capture on falling edge
- 11 Capture on both rising and falling edges

CCISx Bit 13-12 Capture/compare input select. These bits select the TACCRx input signal. See the device-specific data sheet for specific signal connections.
- → 00 CCIxA
- 01 CCIxB
- 10 GND
- 11 V_{CC}

SCS Bit 11 Synchronize capture source. This bit is used to synchronize the capture input signal with the timer clock.
- → 0 Asynchronous capture
- 1 Synchronous capture

```
TA0CCTL1 =
```


Timer A: PWM

EXEMPLO 3

- **4º Passo: Configurar TA0CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP Bit 8 Capture mode
- → 0 Compare mode
- 1 Capture mode

OUTMODx Bits 7-5 Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
- 000 OUT bit value
- 001 Set
- 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- → 111 Reset/set

CCIE Bit 4 Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
- → 0 Interrupt disabled
- 1 Interrupt enabled

```
TA0CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2
```


Timer A: PWM

EXEMPLO 3

- **4º Passo: Configurar TA0CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCI Bit 3 Capture/compare input. The selected input signal can be read by this bit.

OUT Bit 2 Output. For output mode 0, this bit directly controls the state of the output.
- 0 Output low
- → 1 Output high

COV Bit 1 Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software.
- → 0 No capture overflow occurred
- 1 Capture overflow occurred

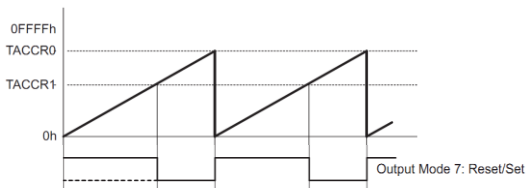
CCIFG Bit 0 Capture/compare interrupt flag
- → 0 No interrupt pending
- 1 Interrupt pending

```
TA0CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;
```

Timer A: PWM

EXEMPLO 3

- **5º Passo:** Calcular os valores para TA0CCR0 e TA0CCR1
  - **TA0CCR0** = ?
  - **TA0CCR1** = ?



Timer A: PWM

EXEMPLO 3

- **Configurações do TA0**

```
void ini_TA0_PWM(void){

    // Fonte de clock: ACLK = 32768 Hz
    // Fator Div.: 1
    // Modo UP sem geracao de int.
    TA0CTL = TASSEL0 + MC0;

    // Modo Reset/set
    // Saida inicializada em nivel alto
    TA0CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;

    TA0CCR0 = 327;  // Para periodo de 10 ms

    TA0CCR1 = 0;    // razao ciclica de 0 % - led apagado
}
```


Timer A: PWM

EXEMPLO 3

- **Configurações do TA1 - Debouncer**

```
void ini_TA1_Debouncer(void){

    // Fonte de clock: SMCLK = 2 MHz
    // Fator Div.: 1
    // Modo Parado (inicial).
    TA1CTL = TASSEL1;

    TA1CCTL0 = CCIE;

    TA1CCR0 = 10000;  // Para tempo de 5 ms

}
```


Timer A: PWM

EXEMPLO 3

- **6º Passo: RTI da Porta 1**

```
#pragma  vector=PORT1_VECTOR
__interrupt void P1_RTI(void){

    P1IFG &= ~BIT3; // Limpa flag - Opcional

    P1IE &= ~BIT3;  // Desabilita int. do BIT3 da P1

    // Timer1 A configurado para o Debouncer de S2

    TA1CTL |= MC0; // Inicia temporizador - TA1
}
```


Timer A: PWM

EXEMPLO 3

- **7º Passo: RTI do Timer 1**

```
#pragma  vector=TIMER1_A0_VECTOR
__interrupt void RTI_Timer1(void){
    TA1CTL &= ~MC0;      // Para o Timer 1

    if( (~P1IN) & BIT3 ) {       // Alterar a largura do pulso
          if(TA0CCR1 >= 328){
             TA0CCR1 = 0;
          }else{
             TA0CCR1 += 82;
          }
    }
    P1IFG &= ~BIT3;      // Limpa flag de int. do BIT3 da P1
    P1IE = BIT3;         // Interrupcao do BIT3 da P1 habilitada
}
```


Timer A: PWM

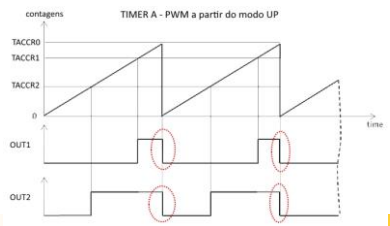
EXEMPLO 3

- **8º Passo:** Código Principal

```
#include <msp430.h>

void config_ini(void);   // Prototipos das funcoes
void ini_P1_P2(void);
void ini_TA0_PWM(void);
void ini_TA1_Debouncer(void);

void main(void) {
    config_ini();
    ini_P1_P2();
    ini_TA0_PWM();
    ini_TA1_Debouncer();

    do{
          _BIS_SR(LPM0_bits + GIE); // Entra no LPM0
    }while(1);
}
```

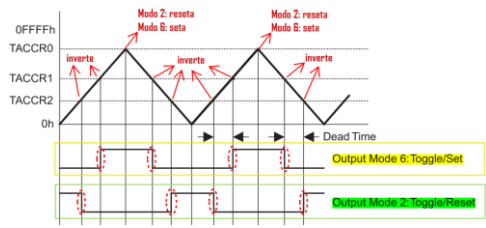
Timer A - PWM

- **Sinais PWM alinhados a BORDA**
  - **Problema:** 2 ou mais comutações ao mesmo instante
    - **Maior Interferência CONDUZIDA e IRADIADA!**
    - **Maior oscilação da tensão de alimentação!**
    - **Efeitos significativo em sistemas com potência elevada...**



Timer A - PWM

- **Sinais PWM alinhados ao CENTRO** (do ciclo de contagem)
  - **MODOS DE SAÍDA** que podem ser usados para gerar PWM alinhado ao centro
    - Modo 2: **Inverter/Resetar**
    - Modo 6: **Inverter/Setar**

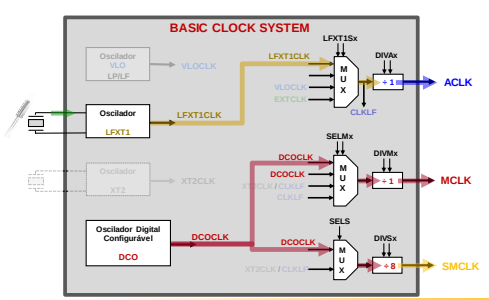


Timer A - PWM

- **EXEMPLO 4**
  - Use o **Timer1 A** do microcontrolador MSP430G2553 para gerar os sinais PWM s_A e s_B alinhados a BORDA e ao CENTRO, com frequência de **204 Hz** e razões cíclicas **RC$_A$** de **50%** e **RC$_B$** de **80%**. Cada vez que **S2** for pressionada deve-se alternar entre PWM alinhado a BORDA e ao CENTRO.
    - **1º Passo:** Função para **inicialização** do uCON
    - **2º Passo:** Função para **inicialização** das portas de I/O
    - **3º Passo:** Função para configuração do **Timer1** para PWM alinhado a BORDA
    - **4º Passo:** Função para configuração do **Timer1** para PWM alinhado ao CENTRO
    - **5º Passo:** Função de inicialização do **Timer0** para o **debouncer** de **S2**
    - **6º Passo:** RTI da Porta 1
    - **7º Passo:** RTI do Módulo 0 do Timer0

Timer A - PWM

- **1º Passo**: Função para inicialização do uCON



Timer A - PWM

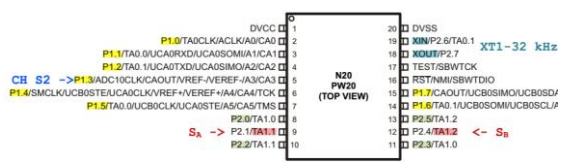
- **1º Passo**: Função para inicialização do uCON

```
void ini_uCon(void){

    WDTCTL = WDTPW | WDTHOLD;    // Para o watchdog timer

    DCOCTL = CALDCO_8MHZ;
    BCSCTL1 = CALBC1_8MHZ;
    BCSCTL2 = DIVS0 + DIVS1;
    BCSCTL3 = XCAP0 + XCAP1;

    while( BCSCTL3 & LFXT1OF);

    __enable_interrupt();

}
```

```
/* CONFIGURACAO do BCS
 * -> Sinais de saida do BCS:
 *      MCLK  -> 8 MHz
 *      SMCLK ->  1 MHz
 *      ACLK  -> 32768 Hz
 * -> Osciladores do BCS
 *    VLO - Nao utilizado
 *    XT2 - Nao esta presente
 *    LFXT1 - xtal 32k
 *    DCO ~ 8 MHz
 */
```


Timer A - PWM

- **2º Passo**: Inicialização das Portas 1 e 2



Timer A - PWM

- **2º Passo**: Inicialização das Portas 1 e 2

Table 20. Port P2 (P2.0 to P2.5) Pin Functions

PIN NAME (P2.x)	x	FUNCTION	CONTROL BITS AND SIGNALS[1]		
			P2DIR.x	P2SEL.x	P2SEL2.x
P2.0/	0	P2.x (I/O)	I: 0; O: 1	0	0
TA1.0/		Timer1_A3.CCI0A	0	1	0
		Timer1_A3.TA0	1	1	0
Pin Osc		Capacitive sensing	X	0	1
P2.1/	1	P2.x (I/O)	I: 0; O: 1	0	0
TA1.1/		Timer1_A3.CCI1A	0	1	0
		Timer1_A3.TA1	1	1	0
Pin Osc		Capacitive sensing	X	0	1
P2.2/	2	P2.x (I/O)	I: 0; O: 1	0	0
TA1.1/		Timer1_A3.CCI1B	0	1	0
		Timer1_A3.TA1	1	1	0
Pin Osc		Capacitive sensing	X	0	1
P2.3/	3	P2.x (I/O)	I: 0; O: 1	0	0
TA1.0/		Timer1_A3.CCI0B	0	1	0
		Timer1_A3.TA0	1	1	0
Pin Osc		Capacitive sensing	X	0	1
P2.4/	4	P2.x (I/O)	I: 0; O: 1	0	0
TA1.2/		Timer1_A3.CCI2A	0	1	0
		Timer1_A3.TA2	1	1	0
Pin Osc		Capacitive sensing	X	0	1

Timer A - PWM

- **2º Passo**: Inicialização das Portas 1 e 2

```
void ini_P1_P2(void){

    P1DIR = ~BIT3;          /* CONFIGURAÇÕES da PORTA 1
    P1OUT = BIT3;           * -> P1.3 - S2: Entrada com resistor de pull-up
    P1REN = BIT3;           * e int. por borda de descida.
    P1IES = BIT3;           * -> P1.x - N.C.: Saidas em nivel baixo.
    P1IFG = 0;              *
    P1IE = BIT3;            * CONFIGURACÇÕES da PORTA 2
                            * -> Pino 9 (P2.1) -> TA1.1
                            * -> Pino 12 (P2.4) -> TA1.2
    P2DIR = 0xFF;           * -> Pinos 18 e 19: mantém funções XIN e XOUT
    P2OUT = 0;              * -> Demais pinos como saida em nivel baixo.
    P2SEL |= BIT1 + BIT4;   */
}
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused Bits 15-10 Unused

TASSELx Bits 9-8 Timer_A clock source select
- 00 TACLK
- 01 ACLK
- → 10 SMCLK
- 11 INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)

IDx Bits 7-6 Input divider. These bits select the divider for the input clock.
- → 00 /1
- 01 /2
- 10 /4
- 11 /8

```
TA1CTL = TASSEL1 +
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

MCx Bits 5-4 Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
- 00 Stop mode: the timer is halted.
- → 01 Up mode: the timer counts up to TACCR0.
- 10 Continuous mode: the timer counts up to 0FFFFh.
- 11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.

Unused Bit 3 Unused

TACLR Bit 2 Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.

TAIE Bit 1 Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
- → 0 Interrupt disabled
- 1 Interrupt enabled

TAIFG Bit 0 Timer_A interrupt flag
- → 0 No interrupt pending
- 1 Interrupt pending

```
TA1CTL = TASSEL1 + MC0;
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx Bit 15-14 Capture mode
- → 00 No capture
- 01 Capture on rising edge
- 10 Capture on falling edge
- 11 Capture on both rising and falling edges

CCISx Bit 13-12 Capture/compare input select. These bits select the TACCRx input signal. See the device-specific data sheet for specific signal connections.
- → 00 CCIxA
- 01 CCIxB
- 10 GND
- 11 V_{CC}

SCS Bit 11 Synchronize capture source. This bit is used to synchronize the capture input signal with the timer clock.
- → 0 Asynchronous capture
- 1 Synchronous capture

```
TA1CCTL1 =
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP Bit 8 Capture mode
- → 0 Compare mode
- 1 Capture mode

OUTMODx Bits 7-5 Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
- 000 OUT bit value
- 001 Set
- 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- → 111 Reset/set

CCIE Bit 4 Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
- 0 Interrupt disabled
- 1 Interrupt enabled

```
TA1CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2
```

Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCI	Bit 3	Capture/compare input. The selected input signal can be read by this bit.
OUT	Bit 2	Output. For output mode 0, this bit directly controls the state of the output.
		0 Output low
		1 Output high
COV	Bit 1	Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software
		0 No capture overflow occurred
		1 Capture overflow occurred
CCIFG	Bit 0	Capture/compare interrupt flag
		0 No interrupt pending
		1 Interrupt pending

```
TA1CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL2**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx	Bit 15-14	Capture mode
		00 No capture
		01 Capture on rising edge
		10 Capture on falling edge
		11 Capture on both rising and falling edges
CCISx	Bit 13-12	Capture/compare input select. These bits select the TACCRx input signal. See the device-specific data sheet for specific signal connections.
		00 CCIxA
		01 CCIxB
		10 GND
		11 V_{CC}
SCS	Bit 11	Synchronize capture source. This bit is used to synchronize the capture input signal with the timer clock.
		0 Asynchronous capture
		1 Synchronous capture

```
TA1CCTL2 =
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL2**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP	Bit 8	Capture mode
		0 Compare mode
		1 Capture mode
OUTMODx	Bits 7-5	Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
		000 OUT bit value
		001 Set
		010 Toggle/reset
		011 Set/reset
		100 Toggle
		101 Reset
		110 Toggle/set
		111 Reset/set
CCIE	Bit 4	Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
		0 Interrupt disabled
		1 Interrupt enabled

```
TA1CCTL2 = OUTMOD0 + OUTMOD1 + OUTMOD2
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCTL2**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCI	Bit 3	Capture/compare input. The selected input signal can be read by this bit.
OUT	Bit 2	Output. For output mode 0, this bit directly controls the state of the output.
		0 Output low
		1 Output high
COV	Bit 1	Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software
		0 No capture overflow occurred
		1 Capture overflow occurred
CCIFG	Bit 0	Capture/compare interrupt flag
		0 No interrupt pending
		1 Interrupt pending

```
TA1CCTL2 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;
```


Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA
  - **Registrador TA1CCR0**
    - Freq. PWM: 204 Hz .:. **TA1CCR0** = (1.000.000 / 204) - 1 = **4.901**
  - **Registrador TA1CCR1**
    - RC S_A = 50 % .:. **TA1CCR1** = (4902*0,5) - 1 = **2.450**
  - **Registrador TA1CCR2**
    - RC S_B = 80 % .:. **TA1CCR2** = (4902*0,8) - 1 ≈ **3.921**

Timer A - PWM

- **3º Passo**: Configuração do Timer1 para PWM alinhado a BORDA

```
void ini_Timer1_PWM_Borda(void){

    TA1CTL = TASSEL1 + MC0;

    TA1CCTL1 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;

    TA1CCTL2 = OUTMOD0 + OUTMOD1 + OUTMOD2 + OUT;

    TA1CCR0 = 4901;

    TA1CCR1 = 2450;

    TA1CCR2 = 3921;
}
```

Timer A - PWM

- **4º Passo**: Configuração do Timer1 para **PWM alinhado ao CENTRO**
  - **Registrador TA1CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused Bits 15-10 Unused

TASSELx Bits 9-8 Timer_A clock source select
- 00 TACLK
- 01 ACLK
- → 10 SMCLK
- 11 INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)

IDx Bits 7-6 Input divider. These bits select the divider for the input clock.
- → 00 /1
- 01 /2
- 10 /4
- 11 /8

```
TA1CTL = TASSEL1 +
```


Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO
  - **Registrador TA1CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

MCx Bits 5-4 Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
- 00 Stop mode: the timer is halted.
- 01 Up mode: the timer counts up to TACCR0.
- 10 Continuous mode: the timer counts up to 0FFFFh.
- → 11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.

Unused Bit 3 Unused

TACLR Bit 2 Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.

TAIE Bit 1 Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
- → 0 Interrupt disabled
- 1 Interrupt enabled

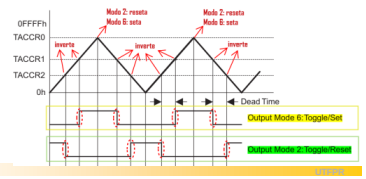
TAIFG Bit 0 Timer_A interrupt flag
- → 0 No interrupt pending
- 1 Interrupt pending

```
TA1CTL = TASSEL1 + MC0 + MC1;
```


Timer A - PWM

- **4º Passo**: Configuração do Timer1 para **PWM alinhado ao CENTRO**
  - **Registrador TA1CCTL1**

OUTMODx Bits 7-5 Output mode. Modes 2
- 000 OUT bit value
- 001 Set
- 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- 111 Reset/set



Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO
  - **Registrador TA1CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP Bit 8 Capture mode
- → 0 Compare mode
- 1 Capture mode

OUTMODx Bits 7-5 Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
- 000 OUT bit value
- 001 Set
- → 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- 111 Reset/set

CCIE Bit 4 Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
- 0 Interrupt disabled
- 1 Interrupt enabled

```
TA1CCTL1 = OUTMOD1 +
```


Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO
  - **Registrador TA1CCTL1**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCI Bit 3 Capture/compare input. The selected input signal can be read by this bit.

OUT Bit 2 Output. For output mode 0, this bit directly controls the state of the output.
- 0 Output low
- → 1 Output high

COV Bit 1 Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software
- → 0 No capture overflow occurred
- 1 Capture overflow occurred

CCIFG Bit 0 Capture/compare interrupt flag
- → 0 No interrupt pending
- 1 Interrupt pending

```
TA1CCTL1 = OUTMOD1 + OUT;
```


Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO
  - **Registrador TA1CCTL2**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP Bit 8 Capture mode
- → 0 Compare mode
- 1 Capture mode

OUTMODx Bits 7-5 Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
- 000 OUT bit value
- 001 Set
- → 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- 111 Reset/set

CCIE Bit 4 Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
- 0 Interrupt disabled
- 1 Interrupt enabled

```
TA1CCTL2 = OUTMOD1 +
```

Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO

  – **Registrador TA1CCTL2**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CCI	Bit 3	Capture/compare input. The selected input signal can be read by this bit.
OUT	Bit 2	Output. For output mode 0, this bit directly controls the state of the output.
	0	Output low
	1	Output high
COV	Bit 1	Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software
	0	No capture overflow occurred
	1	Capture overflow occurred
CCIFG	Bit 0	Capture/compare interrupt flag
	0	No interrupt pending
	1	Interrupt pending

```
TA1CCTL2 = OUTMOD1 + OUT;
```


Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO

  – **Registrador TA1CCR0**

  - Freq. PWM: 204 Hz .:. **TA1CCR0** = (1.000.000 / 204) **/ 2** - 1 = **2.450**

  – **Registrador TA1CCR1**

  - RC S_A = 50 % .:. **TA1CCR1** = (4902*0,5) **/ 2** - 1 = **1.224**

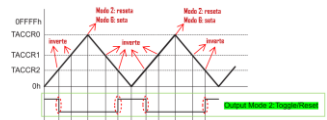
  – **Registrador TA1CCR2**

  - RC S_B = 80 % .:. **TA1CCR2** = (4902*0,8) **/ 2** - 1 ≈ **1.960**

Timer A - PWM

- **4º Passo**: Configuração do Timer1 para PWM alinhado ao CENTRO

```
void ini_Timer1_PWM_Centro(void){

    TA1CTL = TASSEL1 + MC0 + MC1;

    TA1CCTL1 = OUTMOD1 + OUT;

    TA1CCTL2 = OUTMOD1 + OUT;

    TA1CCR0 = 2450;

    TA1CCR1 = 1224;

    TA1CCR2 = 1960;
}
```



Timer A - PWM

- **5º Passo**: de inicialização do Timer0 para o debouncer de S2

  – **Registrador TA0CTL**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused	Bits 15-10	Unused
TASSELx	Bits 9-8	Timer_A clock source select
	00	TACLK
	01	ACLK
	10	SMCLK
	11	INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)
IDx	Bits 7-6	Input divider. These bits select the divider for the input clock.
	00	/1
	01	/2
	10	/4
	11	/8

```
TA0CTL = TASSEL1 +
```


Timer A - PWM

- **5º Passo**: de inicialização do Timer0 para o debouncer de S2

  – **Registrador TA0CTL (contador inicialmente PARADO)**

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

MCx	Bits 5-4	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
	00	Stop mode: the timer is halted.
	01	Up mode: the timer counts up to TACCR0.
	10	Continuous mode: the timer counts up to 0FFFFh.
	11	Up/down mode: the timer counts up to TACCR0 then down to 0000h.
Unused	Bit 3	Unused
TACLR	Bit 2	Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.
TAIE	Bit 1	Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
	0	Interrupt disabled
	1	Interrupt enabled
TAIFG	Bit 0	Timer_A interrupt flag
	0	No interrupt pending
	1	Interrupt pending

```
TA0CTL = TASSEL1;
```


Timer A - PWM

- **5º Passo**: de inicialização do Timer0 para o debouncer de S2

  – **Registrador TA0CCTL0**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CAP	Bit 8	Capture mode
	0	Compare mode
	1	Capture mode
OUTMODx	Bits 7-5	Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
	000	OUT bit value
	001	Set
	010	Toggle/reset
	011	Set/reset
	100	Toggle
	101	Reset
	110	Toggle/set
	111	Reset/set
CCIE	Bit 4	Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag.
	0	Interrupt disabled
	1	Interrupt enabled

```
TA0CCTL0 = CCIE;
```

Timer A - PWM

- **5º Passo**: de inicialização do Timer0 para o debouncer de S2
  - **Registrador TA0CCR0**
    - Tempo debouncer = 5 ms
    - **TA0CCR0** = 1.000.000 * 0,005 = **5.000**

```
void ini_TA0_Debouncer(void){
    TA0CTL = TASSEL1;
    TA0CCTL0 = CCIE;
    TA0CCR0 = 5000;
}
```


Timer A - PWM

- **6º Passo**: RTI da Porta 1

```
#pragma vector=PORT1_VECTOR
__interrupt void RTI_da_Porta_1(void){
    P1IFG &= ~BIT3;
    P1IE &= ~BIT3;
    TA0CTL |= MC0;
}
```


Timer A - PWM

- **7º Passo**: RTI do Módulo 0 do Timer 0

```
#pragma vector=TIMER0_A0_VECTOR
__interrupt void RTI_do_M0_do_Timer_0(void){
   TA0CTL &= ~MC0;
   if( (~P1IN) & BIT3  ){
            if( ctrl == 0 ){
                ctrl = 1; // PWM alinhado a Borda
                TA1CTL = TACLR;
                TA1CCTL1 = 0;
                TA1CCTL2 = 0;
                ini_Timer1_PWM_Borda();
            }else{
                ctrl = 0; // PWM alinhado ao Centro
                TA1CTL = TACLR;
                TA1CCTL1 = 0;
                TA1CCTL2 = 0;
                ini_Timer1_PWM_Centro();
            }
   }
    P1IFG &= ~BIT3;
    P1IE |= BIT3;
}
```


Timer A - PWM

- **EXERCÍCIO PWM**
  - Use o **Timer1 A** do microcontrolador MSP430G2553 para **gerar um sinal senoidal com frequência de 60 Hz** a partir de um sinal PWM.
    - **1º Passo:** Função para **inicialização** do uCON;
    - **2º Passo:** Função para **inicialização** das portas de I/O;
    - **3º Passo:** Função para inicialização do **Timer1** para PWM;
    - **4º Passo:** Criação de um vetor com valores de ajuste de TA1CCR1 a partir da resposta de uma senoide;
    - **5º Passo:** RTI do Módulo 1 do Timer1 para atualização do sinal PWM de acordo com valores de referência de amplitude de uma senoide.

FIM