TIMERS

Timers do ATmega328

- O Atmega328 (utilizado no Arduino UNO) possui 3 timers
- TIMER0 e TIMER2 são de 8 bits (contam de 0 a 255)
- TIMER1é de 16 bits (conta de 0 de 65535)
- Esses temporizadores são importantes para diversas funcionalidades:
  - Temporização;
  - Contagem de eventos externos;
  - Geração de sinais PWM;
  - Interrupções periódicas;
  - Medida de intervalos de pulsos

Timers do ATmega328

- A biblioteca do Arduino abstrai o uso destes temporizadores em muitas de suas funções
- Por exemplo, as funções delay(), millis(), micros(), tone(), analogWrite() utilizam recursos de timers para o funcionamento
- O TIMER1 é utilizado somente em algumas bibliotecas específicas no Arduino, podendo ser utilizado para outras finalidades sem causar muito impacto

Timers do ATmega328

- Existem três maneiras de usar os timers para interrupções:
  - Interrupção de estouro de temporizador (***Timer Overflow Interrupt***)
  - Interrupção por comparação do temporizador (***Timer Compare Interrupt***)
  - Interrupção de captura do temporizador (***Timer Capture Interrupt***)

Interrupção por estouro do temporizador

- O estouro do temporizador é uma condição em que o temporizador contou além do seu número máximo
- Para o Timer0 e Timer2, o estouro ocorre quando a contagem passa de 255 e volta para 0
- Para o Timer1, o estouro ocorre quando a contagem passa de 65535 e volta para 0
- A configuração do bit **TOIE** em cada ***Timer Interrupt Mask Register*** (**TIMSKx**), ativa a interrupção por estouro do temporizador

Interrupção por estouro do temporizador

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
(0x70)	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	TIMSK2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer Interrupt Mask Register

Interrupção por estouro do temporizador

- A fórmula do tempo de estouro é

$$T_{overflow} = \frac{1}{F_{osc}}.2^{bits}.clock\_div$$

- A placa Arduino UNO tem um oscilador de 16 MHz e o **divisor do clock** é 64 por padrão (**esse valor pode ser alterado**)
- Para o **Timer2**, o tempo de estouro do timer com divisão por 64 será

$$T_{overflow} = \frac{1}{16000000}.2^{8}.64 = 0.001024s$$

Interrupção por estouro do temporizador

- A fórmula do cálculo do valor a ser movido para o registrador TCNTx deve ser

$$V_{overflow} = 2^{bits} - \frac{F_{osc}}{\frac{clock\_div}{F_{overflow}}}$$

- Para o caso de usarmos o Timer1, 16 MHz, com divisão de 1024 e Frequência de overflow 1Hz:

$$V_{overflow} = 65536 - \frac{16000000}{\frac{1024}{1}} = 49911 = 0xC2F7$$

Interrupção por estouro do temporizador

- O divisor do clock pode ser ajustado via os três bits menos significativos do registrador **TCCRxB** (*Timer/Counter Control Register B*)

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
(0xB1)	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	TCCR2B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Interrupção por estouro do temporizador

- Exemplo de valores para o divisor do clock (Timer1)

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

Interrupção por estouro do temporizador

- Exemplo de valores para o divisor do clock (Timer2)

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(No prescaling)
0	1	0	clk_{T2S}/8 (From prescaler)
0	1	1	clk_{T2S}/32 (From prescaler)
1	0	0	clk_{T2S}/64 (From prescaler)
1	0	1	clk_{T2S}/128 (From prescaler)
1	1	0	clk_{T2S}/256 (From prescaler)
1	1	1	clk_{T2S}/1024 (From prescaler)

Interrupção por estouro do temporizador

- Quando o temporizador estoura, o vetor de interrupção TIMERx_OVF deve ser chamado pela CPU do ATMega328
- No caso do Timer2, o ISR seria assim:

```
ISR(TIMER2_OVF_vect){
  . . .
}
```

Exemplo

:: Blink do LED via Timer2

```
const int ledPin = 13;
volatile byte state = LOW;

void setup() {
  pinMode(ledPin, OUTPUT);
  //mascara para o bit menos significativo
  //bit TOIE2 habilita interrupcao por overflow
  TIMSK2 = (TIMSK2 & B11111110) | 0x01;
  //mascara para os tres bits menos significativos
  //bits CS22 CS21 CS20 = 111 = clk/1024
  TCCR2B = (TCCR2B & B11111000) | 0x07;
}

void loop() {
  digitalWrite(ledPin, state);
}

ISR(TIMER2_OVF_vect){
  state = !state;
}
```

Interrupção por estouro do temporizador

- O código anterior chama a interrupção a cada 255 incrementos do registrador, sendo que o clock é divido por 1024
- Outra opção é mover um valor para o registrador TCNT2 (há também o TCNT0 e TCNT1)

Bit	7	6	5	4	3	2	1	0	
(0xB2)	TCNT2[7:0]								TCNT2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Exemplo

:: Blink do LED via Timer2

```
const int ledPin = 13;
volatile byte state = LOW;

void setup() {
  pinMode(ledPin, OUTPUT);
  //mascara para o bit menos significativo
  //bit TOIE2 habilita interrupcao por overflow
  TIMSK2 = (TIMSK2 & B11111110) | 0x01;
  //mascara para os tres bits menos significativos
  //bits CS22 CS21 CS20 = 111 = clk/1024
  TCCR2B = (TCCR2B & B11111000) | 0x07;
  TCNT2 = 0X0F;
}

void loop() {
  digitalWrite(ledPin, state);
}

ISR(TIMER2_OVF_vect){
  TCNT2 = 0X0F;
  state = !state;
}
```

Interrupção por comparação de temporizador

Timer Compare Interrupt

Interrupção por comparação de temporizador

- Esta maneira de usar a interrupção do timer do Arduino é comparar a contagem do timer com um valor específico
- Toda vez que a contagem do timer é igual a esse valor, a interrupção ocorre
- Isso é chamado de Interrupção de Comparação de Temporizador
- Ao usar a interrupção de estouro do temporizador, a interrupção é acionada após as contagens e o registrador muda de tudo 1 (255 ou 65535) para tudo zero
- No modo de comparação, a interrupção ocorre para qualquer valor que definimos

Interrupção por comparação de temporizador

- Além disso, você pode comparar o valor do timer a dois valores de comparação, i.e., A e B
- Para comparar com A, o valor do temporizador é comparado ao registrador OCRxA, em que 'x' é o número do temporizador
- Assim, para o Timer2, o registrador é OCR2A
- Se você quiser que a interrupção seja acionada após 128 contagens, o valor de OCR2A deve ser 128
- Para comparar com B, o valor a ser comparado deve ser gravado no registrador OCR2B
- Se você quiser que **outra** interrupção seja acionada na contagem de 200 então o registrador OCR2B deve ter o valor 200

Interrupção por comparação de temporizador

- Para saber qual foi o acionado (comparação A ou comparação B) há vetores de interrupção correspondentes: TIMER2_COMPA e TIMER2_COMPB
- É preciso ativar a interrupção de comparação
- Para tanto, os bits OCIEA e OCIEB no registrador TIMSK2 faz isso

Bit	7	6	5	4	3	2	1	0	
(0x70)	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	TIMSK2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Exemplo
:: Blink do LED via Timer2 (comparação)

```
const int ledPin = 13;
volatile byte state = LOW;

void setup() {
   pinMode(ledPin, OUTPUT);
   TIMSK2 = (TIMSK2 & B11111001) | 0x06;
   TCCR2B = (TCCR2B & B11111000) | 0x07;
   OCR2A = 128;
   OCR2B = 200;
}
```

Exemplo
:: Blink do LED via Timer2 (comparação)

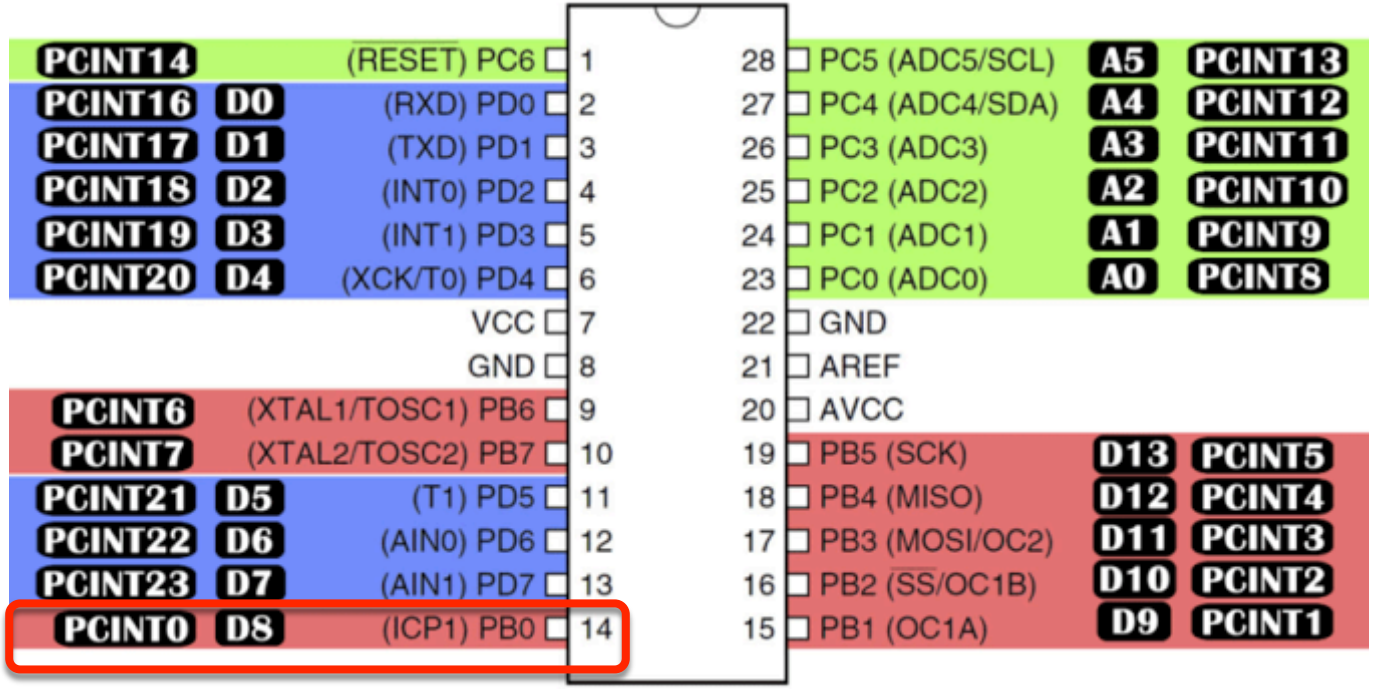
```
void loop() {
    digitalWrite(ledPin, state);
}

ISR(TIMER2_COMPA_vect){
    state = LOW;
}

ISR(TIMER2_COMPB_vect){
    state = HIGH;
}
```

Interrupção de Captura do Temporizador

Timer Capture Interrupt

Interrupção por captura do temporizador

- Esta interrupção só pode ser usada com o Timer1 no Arduino UNO
- Um evento de captura ocorre quando um pulso é lido no pino ICP1 ou no D8



Interrupção por captura do temporizador

- O evento de captura pode ser especificado para saber se o pulso está subindo ou descendo através do **TCCR1B**
- Se o bit **ICES1** estiver definido, o evento de captura ocorrerá a cada borda de subida do pulso
- Caso contrário, a captura ocorre na borda de descida

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TCCR1B – Timer/Counter1 Control Register B

Interrupção por captura do temporizador

- Com relação ao registrador TCCR1B, vamos assumir que a captura acontece a cada borda de subida e não há divisor do clock
- Para acomodar um grande número de pulsos, o TCNT1 é composto por dois registradores de 8 bits, TCNT1L e TCNT1H
- Ou seja, 65535 pulsos no D8 podem ser contados

Interrupção por captura do temporizador

- O valor de TCNT1 é copiado para o registrador ICR1 (***Input Capture Register***) e o flag ICF1, localizado no registrador TIFR1 (***Timer/Counter1 Interrupt Flag Register***), é setado

Bit	7	6	5	4	3	2	1	0	
(0x87)	ICR1[15:8]								ICR1H
(0x86)	ICR1[7:0]								ICR1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Interrupção por captura do temporizador

- Se o bit **ICIE** do registrador **TIMSK1** estiver setado, o vetor de interrupção **TIMER1_CAPT** será executado pela CPU toda vez que ocorrer uma captura

Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Interrupção por captura do temporizador

- Podemos simular a interrupção da captura do temporizador colocando um botão no D8
- Pressionar o botão é como mandar um pulso para este pino
- Usaremos o LED on-board como um indicador visual de que a interrupção foi acionada

Interrupção por captura do temporizador

```
#define ledPin 13
volatile byte state = LOW;

void setup() {
    pinMode(ledPin, OUTPUT);
    pinMode(8, INPUT_PULLUP);
    Serial.begin(9600);
    // habilita interrupcao por timer capture
    TIMSK1 = (TIMSK1 & B11011111) | 0x20;
    // borda de descida e nao divisao do clock
    TCCR1B = (TCCR1B & B10111000) | 0x41;
}
```

Interrupção por captura do temporizador

```
void loop() {
    digitalWrite(ledPin, state);
}

ISR(TIMER1_CAPT_vect){
  state = !state;
}
```

Referências

- https://www.teachmemicro.com/arduino-timer-interrupt-tutorial/
- http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf