Timers do ATmega328 no Arduino

Por

[**Fábio Souza**](https://www.embarcados.com.br/author/fabio-souza/)

 -

13/04/2015

Link : <https://www.embarcados.com.br/timers-do-atmega328-no-arduino>

**ÍNDICE DE CONTEÚDO**[[MOSTRAR](https://www.embarcados.com.br/timers-do-atmega328-no-arduino)]

Introdução

Temporização é fundamental quando se trabalha com circuitos microcontrolados, seja para um simples delay ou para geração de sinais ou eventos periódicos. Desde o mais simples microcontrolador sempre está disponível pelo menos um periférico temporizador/contator. Esse periférico possui hardware dedicado para contagem de tempo e o seu correto uso, auxilia em uma programação eficiente para realização de diversos projetos.

Neste artigo vamos explorar os recursos de um dos timers do ATmega328, o TIMER1, utilizado na plataforma [Arduino](https://www.embarcados.com.br/arduino-primeiros-passos/" \t "_blank), entendendo seu funcionamento e o uso de seus registradores.

Timers do ATmega328

O Atmega328, utilizado na placa [Arduino UNO](https://www.embarcados.com.br/arduino-uno/" \t "_blank), possui 3 timers, sendo dois de 8 bits (TIMER0 e TIMER2) e um de 16 bits (TIMER1). Esses temporizadores são importantes para diversas funcionalidades, tais como:

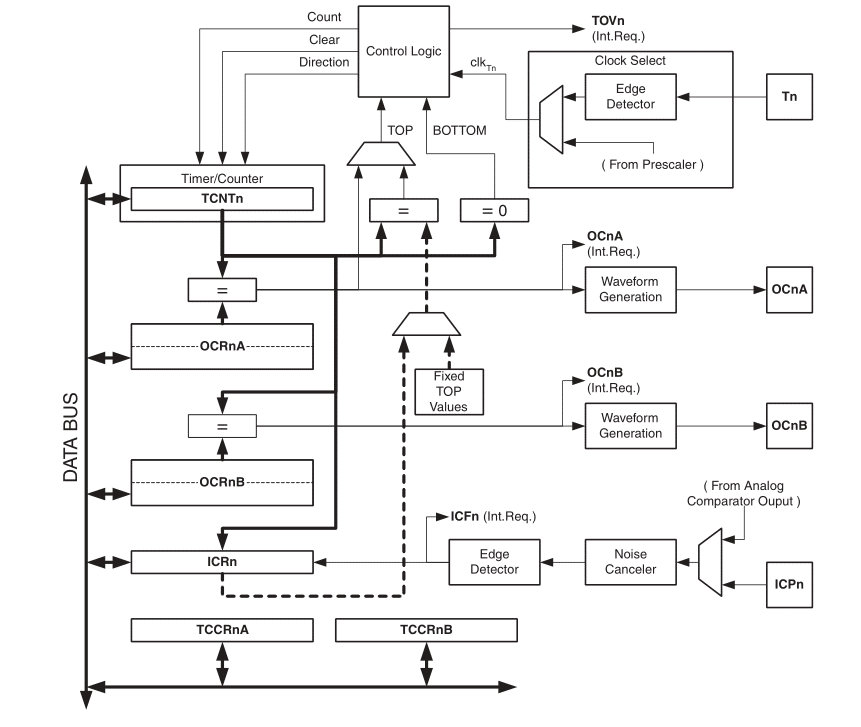
* Temporização;
* Contagem de eventos externos;
* Geração de sinais PWM;
* Interrupções periódicas;
* Medida de intervalos de pulsos.

Cada temporizador possui características próprias e são utilizados conforme os recursos disponíveis. A biblioteca do Arduino abstrai o uso destes temporizadores em muitas de suas funções. Por exemplos, as funções *delay()*, *millis()*, *micros()*, *tone()*, *analogWrite()* utilizam recursos de timers para o funcionamento.

O TIMER1 é utilizado somente em algumas bibliotecas no Arduino específicas, podendo ser utilizado para outras finalidades sem causar muito impacto no funcionamento do restante das funções. A seguir serão apresentadas suas características.

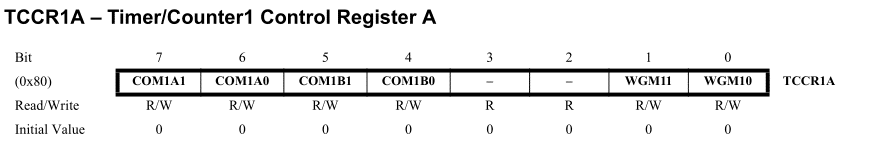
TIMER 1

O TIMER1 é um temporizador de 16 bits que permite a contagem de eventos, geração de sinal PWM, medida de pulsos, etc. Seu diagrama de blocos, conforme apresentado no [datasheet do ATmega328](http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf" \t "_blank)  é apresentado na figura 1:

[](https://www.embarcados.com.br/wp-content/uploads/2015/04/timers-do-atmega328-diagrama-de-blocos-timer1.png)Figura 1 - Diagrama de blocos do TIMER 1 do ATmega328

Registradores

O controle do modo de operação do TIMER1 é feito nos registradores TCCR1A e TCCR1B, conforme descrição a seguir.

[](https://www.embarcados.com.br/wp-content/uploads/2015/04/timers-do-atmega328-tccr1.png)Figura 2 - Registrador TCCR1A

**Bits 7:6** **- COM1A1:0: Compare Output Mode for Channel A**

**Bits 5:4** **– COM1B1:0: Compare Output Mode for Channel B**

COM1A1:0 e COM1B1:0 controlam os pinos de Output compare (OC1A e OC1B), respectivamente, conforme tabelas a seguir:

Tabela 1: Configuração para Modo não PWM

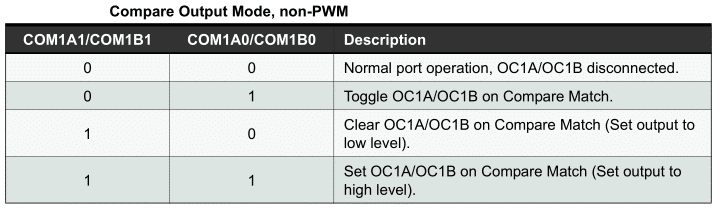
****

Tabela 2: Configuração para modo PWM rápido

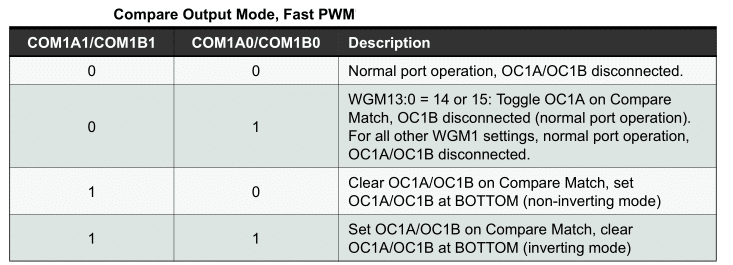
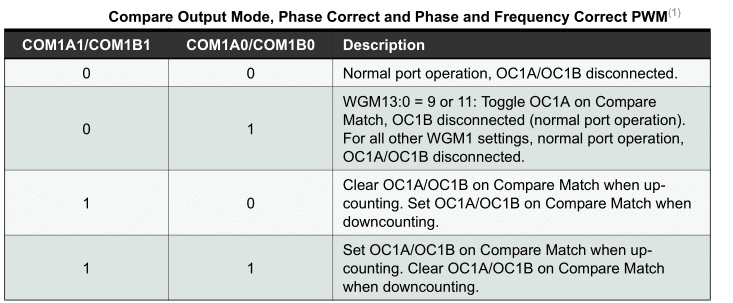


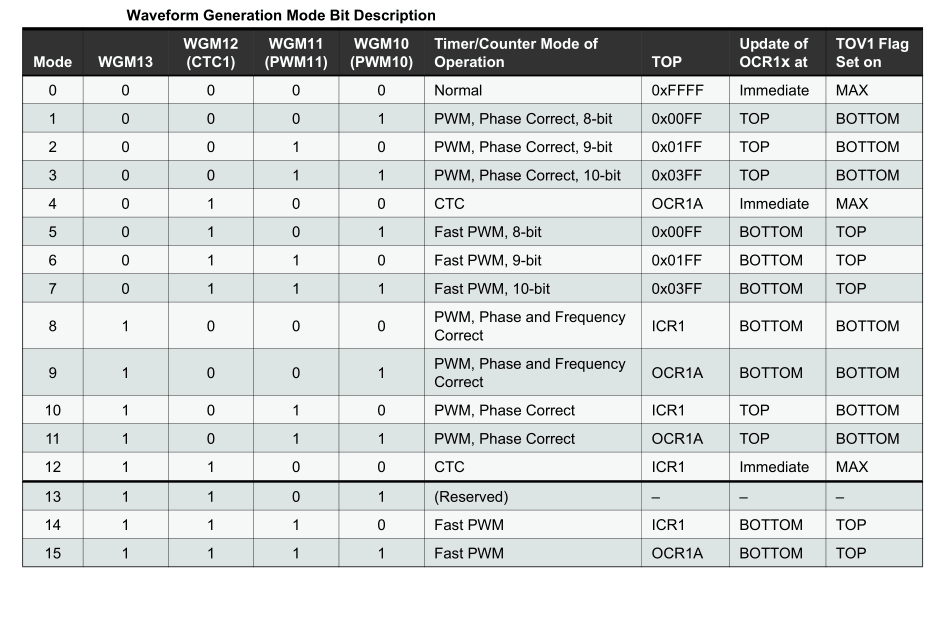
Tabela 3: configuração para modo PWM com correção de fase e frequência

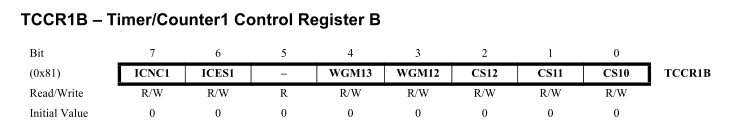


**Bits 1:0**- **WGM11:0: Waveform Generation Mode**

Juntos com os bits WGM13:2 encontrados no registrador TCCR1B, controlam o funcionamento do TIMER1, conforme tabela 4 a seguir:

Tabela 4: Modos de funcionamento do TIMER 1





**Bit 7 – ICNC1: Input Capture Noise Canceler**

Bit para habilitar filtro de ruído no pino de captura ICP1.

**Bit 6 – ICES1: Input Capture Edge Select**

Seleciona qual borda no pino de entrada (ICP1) será usada para disparar evento de captura.

**Bit 5 – Reserved**

Bit reservado, deve ser escrito zero.

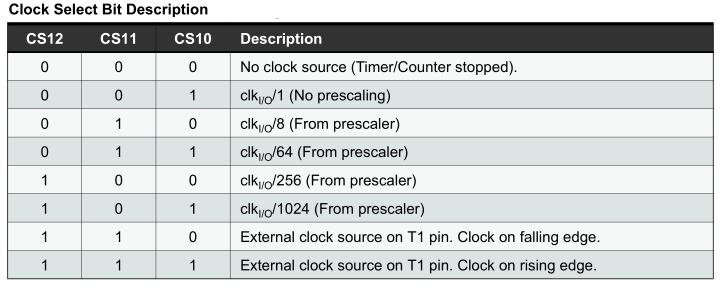
**Bit 4:3 – WGM13:2: Waveform Generation Mode**

Conforme apresentado anteriormente e exibido na tabela 4

**Bit 2:0 – CS12:0: Clock Select**

Bits para seleção de clock, conforme tabela 5

Tabela 5 - Configuração para seleção de clock



Conforme observado na figura 1, exitem outros registradores para funcionamento do TIMER1:

**TCNT1H and TCNT1L – Timer/Counter1:**

Resgistradores de armazenamento de contagem do timer.

**OCR1AH and OCR1AL – Output Compare Register 1 A**

**OCR1BH and OCR1BL – Output Compare Register 1 B:**

Registradores para comparação de contagem com o TCNT1. A igualdade pode gerar uma interrupção ou gera uma saída de onda nos pinos OC1A ou OC1B.

**ICR1H and ICR1L – Input Capture Register 1:**

Registradores para armazenamento de captura quando um evento ocorrer no pino ICP1.

**TIMSK1 – Timer/Counter1 Interrupt Mask Register:**

Registrador para habilitar as interrupções disponíveis no TIMER1.

**TIFR1 – Timer/Counter1 Interrupt Flag Register**

Registrador para flags de interrupções.

Para mais detalhes do funcionamento dos registradores, acesse o [datasheet do ATmega328](http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf" \t "_blank).

Utilizando o TIMER1  no Arduino

**Modo Normal com interrupção por overflow**

O exemplo a seguir exibe como piscar o LED no pino 13, da placa Arduino UNO, em intervalos de 1 segundo e utilizando interrupção por estouro timer:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | #define ledPin 13    void setup()  {    pinMode(ledPin, OUTPUT);      // Configuração do timer1    TCCR1A = 0;                        //confira timer para operação normal pinos OC1A e OC1B desconectados    TCCR1B = 0;                        //limpa registrador    TCCR1B |= (1<<CS10)|(1 << CS12);   // configura prescaler para 1024: CS12 = 1 e CS10 = 1      TCNT1 = 0xC2F7;                    // incia timer com valor para que estouro ocorra em 1 segundo                                       // 65536-(16MHz/1024/1Hz) = 49911 = 0xC2F7      TIMSK1 |= (1 << TOIE1);           // habilita a interrupção do TIMER1  }      void loop()  {    //loop principal. a manipulação do led é feita na ISR  }      ISR(TIMER1\_OVF\_vect)                              //interrupção do TIMER1  {    TCNT1 = 0xC2F7;                                 // Renicia TIMER    digitalWrite(ledPin, digitalRead(ledPin) ^ 1); //inverte estado do led  } |

Nesse exemplo o timer foi configurado para modo normal, com pinos OC1A e OC1B desconectados (TCCR1A = 0). Foi selecionado o prescaler para 1024 através do registrador TCCR1B.  Para que o timer estoure a cada segundo é necessário iniciar seu valor com a diferença entre o seu valor máximo (65536) e o período desejado. O período é calculado levando em consideração a frequência do oscilador e o prescaler selecionado, além da frequência de interrupção desejada. Por fim foi habilitada a interrupção de estouro do TIMER1 através do bit T0IE1 do registrador TIMSK1. A inversão do LED é feita na rotina de interrupção, note que é necessário recarregar o timer para a correta contagem.

**MODO CTC interrupção por comparação**

O exemplo a seguir exibe como piscar o LED, utilizando o modo CTC e gerando interrupção por comparação:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | #define ledPin 13    void setup()  {    pinMode(ledPin, OUTPUT);      // Configuração do TIMER1    TCCR1A = 0;                //confira timer para operação normal    TCCR1B = 0;                //limpa registrador    TCNT1  = 0;                //zera temporizado      OCR1A = 0x3D09;            // carrega registrador de comparação: 16MHz/1024/1Hz = 15625 = 0X3D09    TCCR1B |= (1 << WGM12)|(1<<CS10)|(1 << CS12);   // modo CTC, prescaler de 1024: CS12 = 1 e CS10 = 1    TIMSK1 |= (1 << OCIE1A);  // habilita interrupção por igualdade de comparação  }      void loop()  {    //loop principal. a manipulação do led é feita na ISR  }    ISR(TIMER1\_COMPA\_vect)          // interrupção por igualdade de comparação no TIMER1  {    digitalWrite(ledPin, digitalRead(ledPin) ^ 1);   //inverte estado do LED  } |

Nesse exemplo é utilizado o modo CTC, dessa forma o valor de contagem do timer é constantemente comparado com o registrador OCR1A. Para o funcionamento foi selecionado o modo CTC, bit WGM12 = 1 e configurado o prescaler para 1024. O valor de comparação foi carregado no registrador OCR1A e por último foi habilitada a interrupção de comparação. A inversão do LED é feita na rotina de interrupção, e note que não é necessário reiniciar o timer com um valor, pois o modo CTC zera o timer quando atingido valor de comparação.

Está disponível no sites do Arduino uma biblioteca para manipulação do [TIMER1](http://playground.arduino.cc/Code/Timer1). Essa biblioteca abstrai todas as configurações dos registradores facilitando o uso do TIMER1.