CENTRO UNIVERSITÁRIO DE ANÁPOLIS

UNIEVANGÉLICA

CURSOS SUPERIORES DE COMPUTAÇÃO

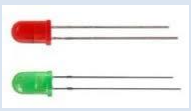
Professor: Alexandre Moraes Tannus - 2018

Arduino: Interrupções

1. OBJETIVOS:

- Conhecer os fundamentos de interrupções
- Implementar interrupções no Arduino

2. MATERIAIS:

	Uma placa Arduino Uno
	LEDs
	Resistores
	*Push-Button*

3. PARTE TEÓRICA

3.1. Interrupções no Arduino

Interrupções são ações que param o fluxo normal de execução de um programa (função loop) e executam uma função que possui prioridade, também conhecida como rotina de serviço de interrupção (ISR – *Interruption Service Routine*). Estas interrupções acionam pinos específicos do microcontrolador nomeados INT (Figura 1).

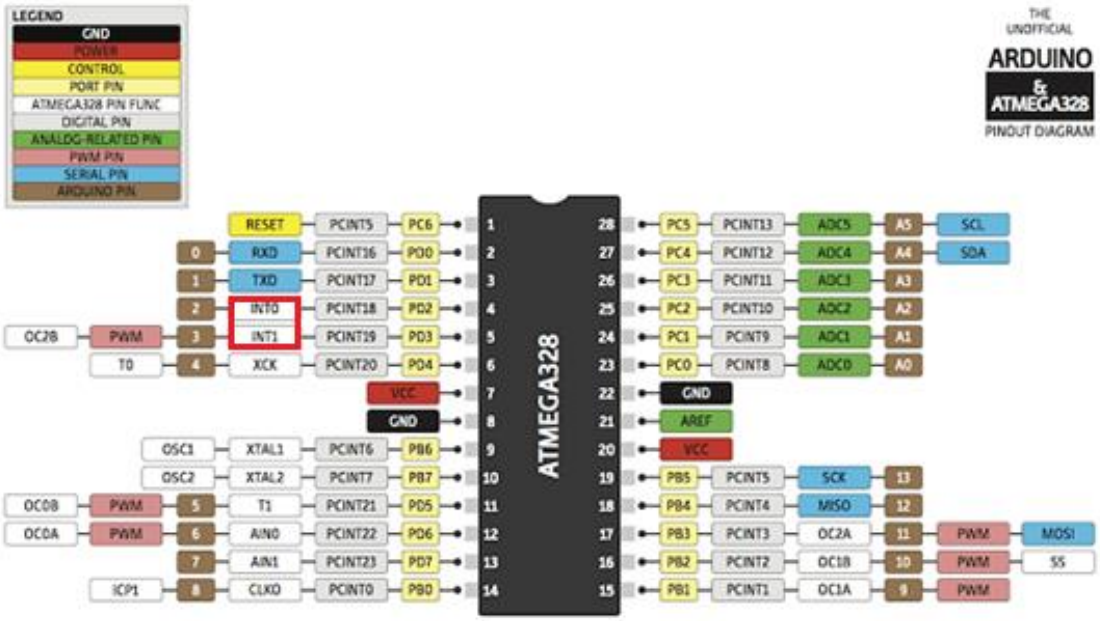


Figura 1 - Pinagem Arduino - Interrupções

Para acionar uma ISR é necessário declarar, através da função $attachInterrupt()$ qual será o número da interrupção utilizada, a rotina de interrupção e a condição de ativação. A sintaxe do comando é:

$$attachInterrupt(interrupcao, ISR, modo)$$

O primeiro parâmetro indica o número da interrupção que será utilizada. Cada placa de Arduino possui uma quantidade diferente de pinos habilitados para interrupções externas, conforme mostra a Tabela 1.

Tabela 1 - Pinos de interrupção de diferentes placas Arduino

Placa	0	1	2	3	4	5
Uno	D2	D3	-	-	-	-
Leonardo	D3	D2	D0	D1	D7	-
Mega 2560	D2	D3	D21	D20	D19	D18

É possível obter o número da interrupção através da função $digialPinToInterrupt(pino)$ que retorna o valor da interrupção para um determinado pino. O uso dessa função é recomendado para facilitar o desenvolvimento e futuro entendimento do código.

A ISR é uma função criada pelo usuário que será executada sempre que a interrupção for ativada. Esta função não possui parâmetros de entrada e nem retorno (void). Esta rotina não permite o uso de $delay$ ou qualquer outra função que interrompa o fluxo. Além disso, o valor obtido através da função $millis$ não é incrementado dentro de uma ISR. Entretanto, caso seja necessário utilizar uma pausa, a função $delayMicroseconds$ pode ser utilizada. O uso de operações em interfaces seriais também não é recomendado, visto que a comunicação serial também gera uma interrupção. Outro ponto importante é que toda variável passível de alteração na ISR deve ser declarada como $volatile$.

O último parâmetro da função $attachInterrupt$ é o modo de ativação da interrupção. Quatro modos são utilizados: *low*, *change, rising* e *falling*. O uso de cada um deles é mostrado na

Tabela 2 - Modos de ativação

Modo	Ativação
LOW	Sempre que a porta estiver em nível baixo
CHANGE	Quando houver qualquer tipo de transição na porta
RISING	Em caso de transição positiva
FALLING	Em caso de transição negativa

4. PARTE PRÁTICA

4.1. Prática 01 – Interrupção

Esta prática introduz a programação de interrupções. Neste experimento a interrupção será associada a um botão. Cada clique no botão incrementará uma variável e desligará o LED da placa (pino 13), que será mostrada na interface serial. Para realizar esta prática monte o circuito da Figura 2 e digite o *sketch* presente no **Erro! Fonte de referência não encontrada.**

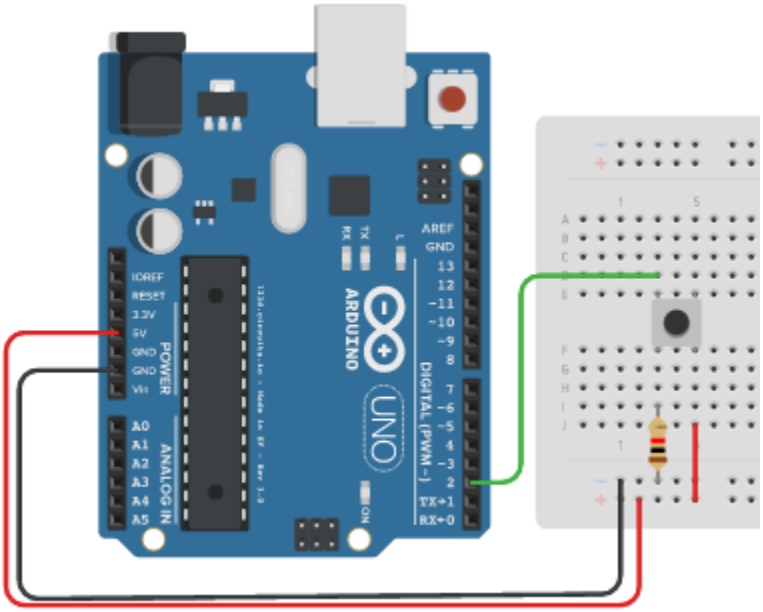


Figura 2 – Circuito da Prática 01

```
#define LED 13

volatile int k;

void setup( ) {
   pinMode(LED, OUTPUT);
   attachInterrup(0, interrupcao
   Serial.begin(9600);
}

void loop( ) {
   digitalWrite(LED, HIGH);
   Serial.println(k);
}

void interrupcao( ) {
   digitalWrite(LED, LOW);
   k++;
}
```

Código 1 - Sketch da Prática 01

4.2. Prática 02 – Desafio

Nesta prática deve ser montado um circuito envolvendo um botão gerador de interrupção, um display de 7 segmentos, um LED e um servomotor . O circuito deve funcionar da seguinte maneira:

Um contador de 0 a 9 funciona continuamente e é mostrado no *display* de 7 segmentos.. O acionamento do botão gera uma interrupção, cuja ISR deverá mapear o valor atual do contador para um valor PWM para o LED e um ângulo para o motor.

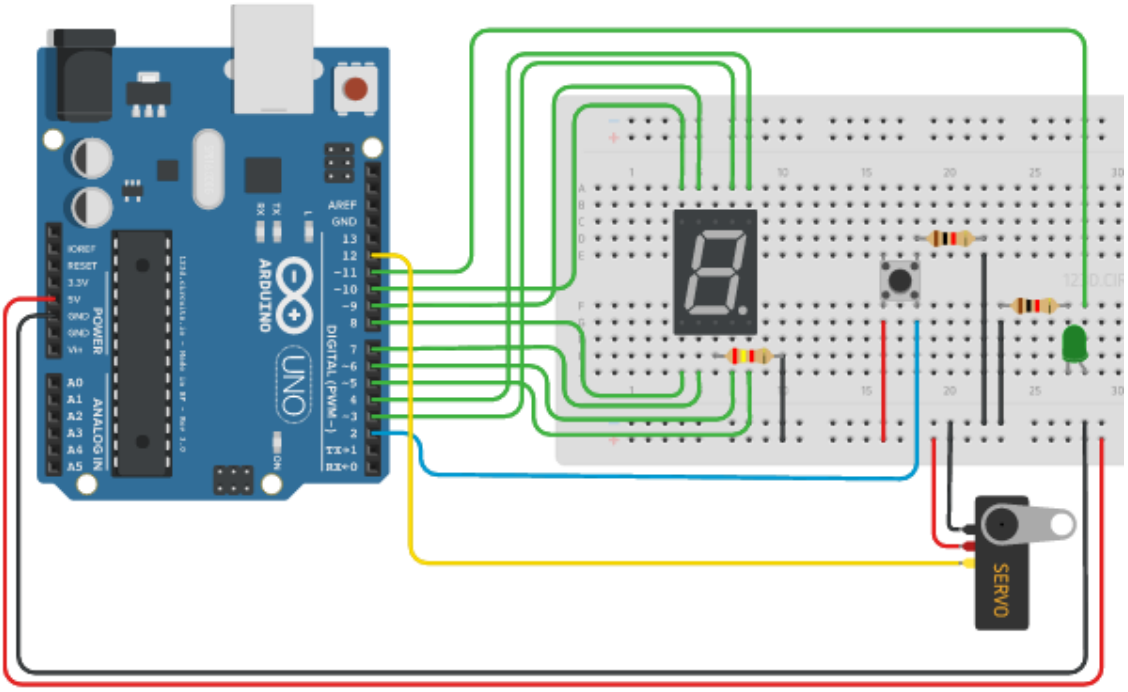


Figura 3 - Circuito da Prática 02

5. REFERÊNCIAS

BANZI, Massimo. *Getting Started with Arduino*. 2ª ed. Sebastopol: O'Reilly, 2011.

EVANS, Martin; NOBLE, Joshua; HOCHENBAUM, Jordan. *Arduino em Ação*. 1ª ed. [S.l.]: Novatec, 2013.

MONK, Simon. *Programação com Arduino: começando com Sketches*. 1ª ed. Porto Alegre: Bookman, 2013.