

Atividade_05_Int - Livro AVR e Arduino – Técnicas de Projeto

Capítulo: 6 (Interrupções)

Título: Usando interrupções externas – INT0 e INT1 e PCINTs (Pin Change Interrupts)

Objetivos: Aprender a ativar e desativar as interrupções externas nos microcontroladores da Atmel

Nesta prática utilizaremos o Tinkercad para simular um circuito simples usando o microcontrolador Atmega328, utilizado nas placas Arduino UNO. Desta vez, programaremos usando um código C para acender e apagar LEDs usando interrupções externas.

1. Procedimentos:

1. Acesse sua conta no Tinkercad (tinkercad.com) e vá para a aba circuits (<https://www.tinkercad.com/circuits>).
2. Você deve fazer um circuito capaz de ler três botões, ligados nas portas PB1, PC3 e PD7. Três LEDs devem ser colocados no circuito. Cada LED troca de estado (ligado → desligado e desligado → ligado) ao toque de um botão em particular. Um starter code é fornecido, mas note que ele usa as portas PORTC5:7.
3. Você deve usar interrupções por mudança de pino (PCINTs) para detectar o clique dos botões, como no starter code. Você deve **eliminar o delay de 200ms** dentro da rotina de interrupção do starter code, para que seu projeto seja avaliado. Note que algo deve ser feito neste caso, já que a interrupção pode ser ativada mais de uma vez no mesmo clique.
4. Agora você deve ligar um botão a INT0 e ligar as interrupções na borda de descida no pino associado. Este botão deve apagar todos os LEDs, caso estejam apagados, deve acender todos os LEDs.

5. Cole o código fonte do microcontrolador ao final deste arquivo e inclua a imagem de seu design. Importante: Deixe seu circuito público no Tinkercad e cole o link para ele aqui:

LINK: https://www.tinkercad.com/things/2LgaucrODfN-brave-tumelo-lahdi/editel?sharecode=SG6fMshmh1PO3j5vrOpXVnMWq9kUjZ_a0eQtBNq3jTM

CÓDIGO:

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

#define F_CPU 16000000UL

#define tst_bit(Y, bit_x)  (Y & (1 << bit_x))
#define set_bit(Y, bit_x)  (Y |= (1 << bit_x))
#define clr_bit(Y, bit_x)  (Y &= ~(1 << bit_x))
#define cpl_bit(y, bit)    (y ^= (1 << bit))

#define LED0 PD5
#define LED1 PD4
#define LED2 PD3

//-----
ISR(PCINT0_vect);
ISR(INT0_vect);
```

```

int main(){

    UCSR0B = 0x00;

    // Pull-up
    DDRB  |= 0x00;
    PORTB |= 0xFF;

    DDRC  |= 0x00;
    PORTC |= 0xFF;

    DDRD |= 0b00111000;
    PORTD |= 0b10000100;

    //habilita as interrupções dos PCINTs
    PCICR |= (1<<PCIE0)|(1<<PCIE1)|(1<<PCIE2);

    //habilita os pinos específicos para interrupções
    PCMSK0 |= (1<<PCINT1);
    PCMSK1 |= (1<<PCINT11);
    PCMSK2 |= (1<<PCINT23);

    //habilita o INT0 na borda de descida
    EICRA |= (1<<ISC01);

    //habilita o INT0 para interrupção
    EIMSK |= (1<<INT0);

    sei();
    //habilita as interrupções

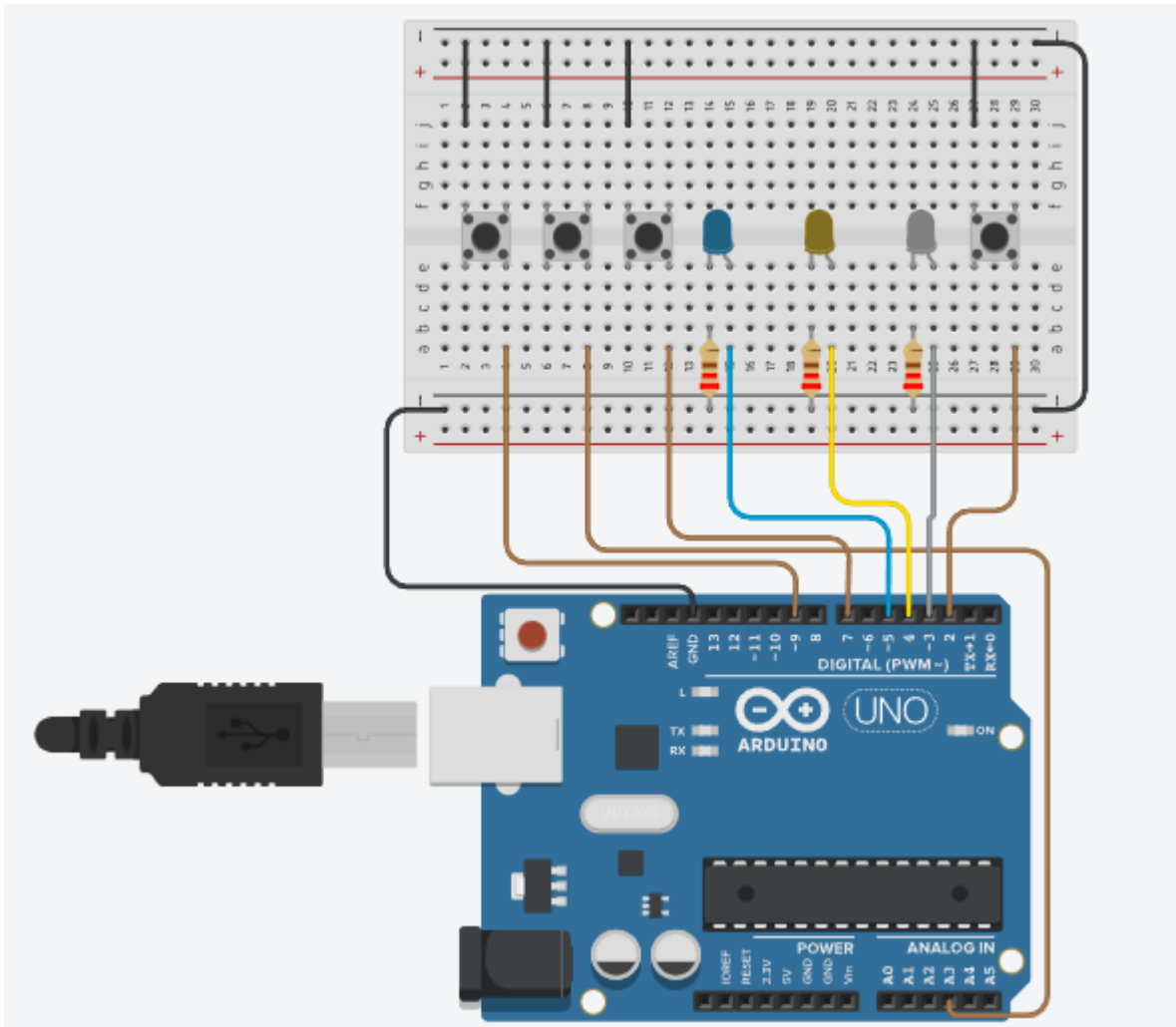
    while(1) {};
    return 1;
}

// Botoes unitarios dos leds
ISR(PCINT0_vect) {
    if(!tst_bit(PINB, PB1))
        cpl_bit(PORTD, LED0);
    else if(!tst_bit(PINC, PC3))
        cpl_bit(PORTD, LED1);
    else if(!tst_bit(PIND, PD7))
        cpl_bit(PORTD, LED2);
}

// botao unico dos leds
ISR(INT0_vect) {
    if (
        !tst_bit(PORTD, PD3)
        && !tst_bit(PORTD, PD4)
        && !tst_bit(PORTD, PD5)
    ){
        set_bit(PORTD, PD3);
        set_bit(PORTD, PD4);
        set_bit(PORTD, PD5);
    }
    else {
        clr_bit(PORTD, PD3);
        clr_bit(PORTD, PD4);
        clr_bit(PORTD, PD5);
    }
}

```

ARDUINO:



ATENÇÃO: Documente seu código. Cada linha/bloco deve deixar explícito o seu papel.

ATENÇÃO: Na versão final do seu projeto, as funções `pinMode()`, `digitalWrite()` e `digitalRead()` são proibidas. O uso delas fará a nota atribuída ser zero.

RÚBRICA:

Utilização do PCINT corretamente: 50%

Utilização do INT corretamente: 50%

Valor desta atividade na média: 0.4