

Atividade Prática

Animação *pushLeds* em Assembly

1 Critérios Avaliativos

Esta atividade deve ser postada em um arquivo compactado **em formato ZIP** no SIGAA impreterivelmente até às ____:____:____ do dia ____/____/____. Recomenda-se a postagem da atividade o quanto antes, de modo a evitar problemas associados à indisponibilidade de acesso à Internet.

O arquivo zip deve conter:

1. O arquivo *workspace* e o diretório *.vscode* do Visual Studio Code.
2. Todos os códigos-fonte (.c, .cpp, .h, .hpp, .s, .inc, .asm) necessários para a compilação com sucesso do código.
3. O arquivo de simulação do Proteus, compatível com a versão 8.6 SP2 build 23525.

A nota do atividade prática é baseada no funcionamento e na qualidade da solução apresentada. São critérios indispensáveis:

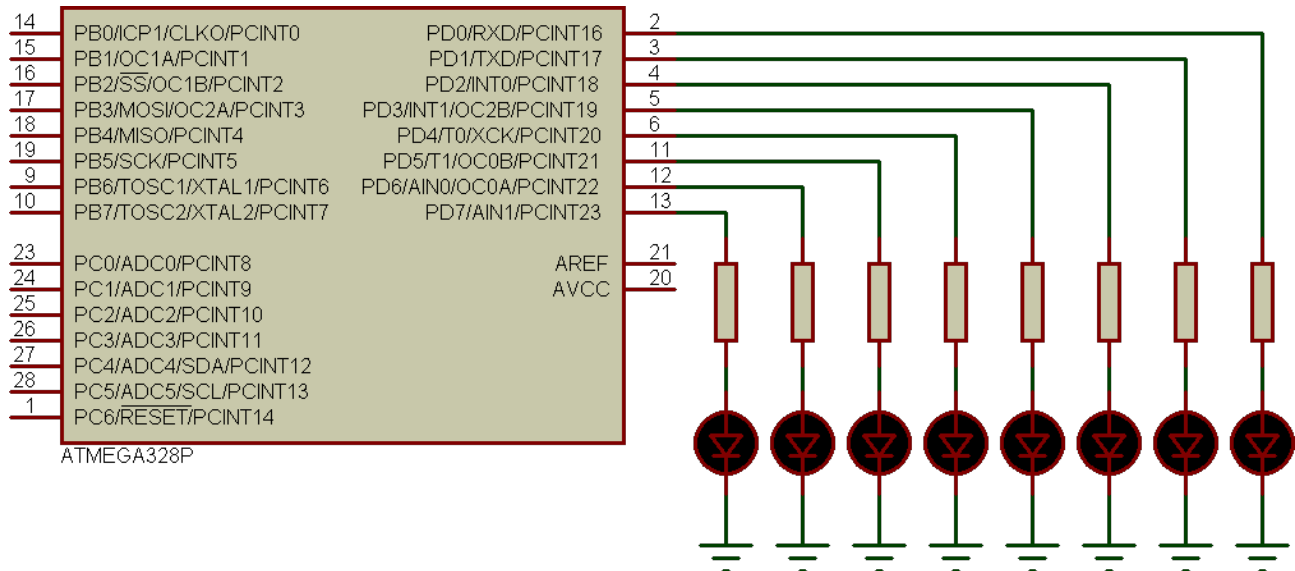
- Funcionamento da solução;
 - Cumprimento dos critérios estabelecidos.
- Organização do código;
 - Documentação (comentários necessários no código-fonte);
 - Endentação coerente;
 - Formatação adequada.
- Otimização;
 - Uso apropriado dos periféricos;
 - Uso racional de memórias RAM, EEPROM e Flash.

2 Objetivo

Criar uma animação estilo *pushLEDs* em linguagem Assembly, utilizando as instruções de comparação, deslocamento e lógica bit a bit.

3 Hardware

A seguir, é apresentado o diagrama esquemático do arquivo de simulação e a lista de componentes e suas configurações:



- Microcontrolador;
 - Device: ATMEGA328P;
 - Library: AVR2;
 - Part Reference: oculto;
 - Part Value: oculto;
 - Encapsulamento: SPDIL28;
 - Fuse CLKDIV8: (1) Unprogrammed;
 - Fuse CLKSEL: (1111) External crystal 8.0–MHz;
 - Clock Frequency: 16 MHz;
- Resistores;
 - Device: RES;
 - Library: DEVICE;
 - Part Reference: oculto;
 - Part Value: oculto;
 - Encapsulamento: RES40;
 - Model Type: DIGITAL;
 - Resistance: 1k5;
- LED;
 - Device: LED-GREEN¹
 - Library: ACTIVE;
 - Part Reference: oculto;

¹Também são aceitos os componentes LED-AQUA, LED-BLUE, LED-ORANGE, LED-PINK, LED-PURPLE, LED-RED, LED-WHITE e LED-YELLOW. Mantendo o posicionamento dos componentes, o aluno está livre para usar a criatividade!

- Part Value: oculto;
- LISA Model File: Digital;
- Full drive current: 1mA;

4 Requisitos Obrigatórios

1. O código deve ser programado para o microcontrolador ATmega328P;
2. O código deve ser programado em C.
3. A animação a ser implementada é do tipo "*push-LEDs*", e funciona da seguinte forma:
 - (a) Considere que os LEDs são enumerados de 7 a 0;
 - (b) Um LEDs é chamado de *pushed* e um LED é chamado de *pusher*;
 - (c) Somente os LEDs *pusher* e *pushed* devem ser acessos;
 - (d) Inicialmente, o LED 0 é *pusher* e o LED 1 é *pushed*;
 - (e) A cada *frame* da animação, o LED *pusher* é incrementado até que ele se sobreponha ao LED *pushed*;
 - (f) Quando os LEDs *pusher* e *pusher* tornam-se coincidentes, o LED imediatamente à esquerda é aceso e torna-se o novo LED *pushed*;
 - (g) O LED *pusher* volta a ser o LED 0 e a animação continua a partir do passo do item 3e até que o LED *pushed* seja explido do conjunto de LEDs, quando, então, a animação reinicia do item 3d.
4. A duração (ou seja, o atraso) de cada passo da animação (*frame*) é de 250ms;
5. Com exceção do primeiro *frame* do *loop* de animação, a lógica envolvida na formação de todos os outros *frames* deve, **obrigatoriamente**, utilizar operações de manipulação bit a bit, sendo **proibido** o uso da instrução de carregamento de imediato (LDI).

5 Sugestões de Implantação




































Os itens a seguir são apenas sugestões, sendo, portanto, sua adoção **facultativos**. Sua implementação é, no entanto, **aconselhada**, seja por apresentarem dicas de implementação ou para aumento da robustez e/ou portabilidade do código final.

- Utilize a convenção de uso de registradores do GCC, ou seja:
 - R0 e R1 são registradores **fixos**: não devem ser modificados em parte alguma do código;
 - R2-R17 e R28-R29 são registradores **recuperáveis**: recomenda-se que, se modificados, tenham seu valor salvo no início e restaurado no final da subrotina. Este processo é chamado de *manutenção de contexto*.
 - R18-R27 e R30-R31 são registradores **descartáveis**: qualquer subrotina pode modificá-los sem necessidade de manutenção de contexto.
 - Se for necessário que um registrador descartável mantenha seu valor prévio após a chamada e retorno de uma subrotina, ele deve ser armazenado na pilha antes da chamada da função e carregado da pilha após o retorno.
 - Considere que o registrador SREG é um registrador descartável.

6 Resultado Desejado

Cada um dos *frames* da animação é apresentado na tabela a seguir. Após o *frame* 35, a animação reinicia a partir do *frame* 1.

Tabela 1: Sequência de *frames* da animação.

Frame	LEDs	Frame	LEDs
1		19	
2		20	
3		21	
4		22	
5		23	
6		24	
7		25	
8		26	
9		27	
10		28	
11		29	
12		30	
13		31	
14		32	
15		33	
16		34	
17		35	
18		1	