EA871 – Laboratório de Programação Básica de Sistemas Digitais

Atividade 05

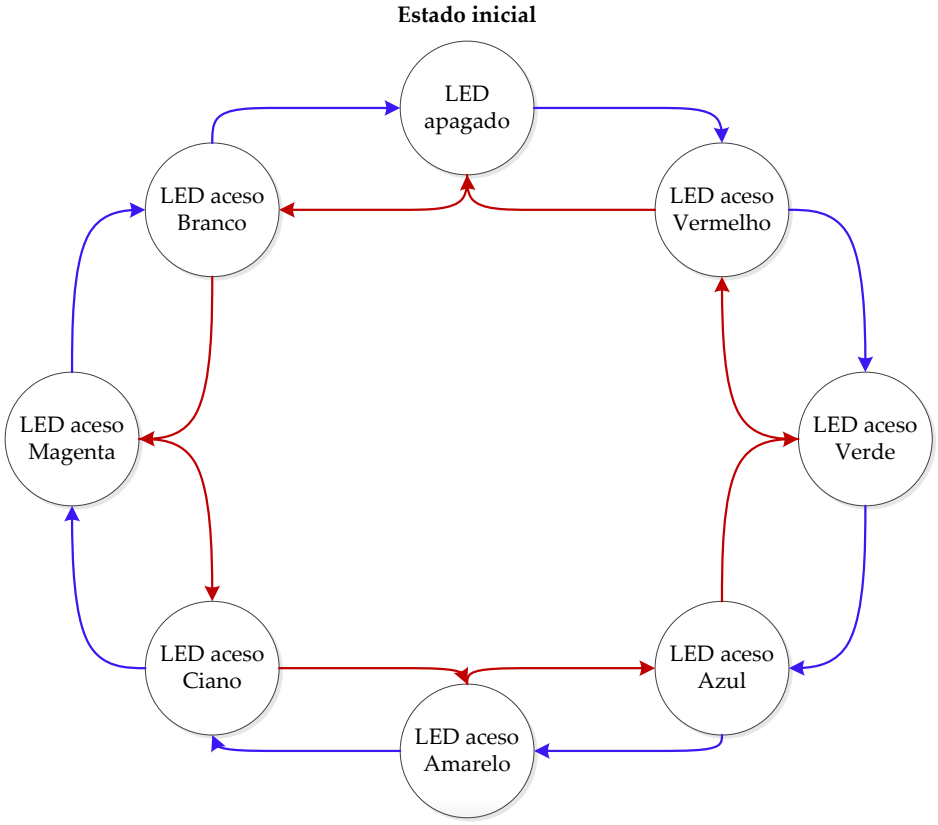
Profs. Levy Boccato, Rafael Ferrari e Tiago Tavares – 2º semestre de 2019

1. Objetivos

- Explorar os diferentes mecanismos de interrupção disponíveis no microcontrolador ATMega328P para lidar com eventos externos.
- Desenvolver um projeto integrando código em linguagem C e *Assembly*.

2. Resumo da Atividade

O desafio proposto nesta atividade é desenvolver um programa que controle um LED RGB, sendo sensível a comandos externos dados por dois botões. Em princípio, o estado do LED RGB será alterado de forma periódica pelo microcontrolador, passando por uma sequência pré-definida de estados. A figura abaixo resume o comportamento desejado para o sistema na forma de uma máquina de estados finitos.



O fluxo dos estados pode se dar no sentido horário (setas em azul) ou no sentido anti-horário (setas em vermelho), sendo que o LED RGB deve permanecer em cada estado e, portanto, em cada uma das possíveis cores, por um intervalo de tempo controlado pelo programa.

Um dos botões, aqui denominado de **botão A**, terá efeito sobre a velocidade de transição entre os estados. Cada vez que ele for pressionado, o intervalo de tempo que o LED permanece em cada estado, i.e., em cada cor, deve ser reduzido pela metade até atingir um valor mínimo (e.g., 0,125 s). Caso o usuário tente reduzir ainda mais esse tempo, ele deve ser reinicializado para o tempo máximo de 1 segundo (que, a propósito, também é o tempo inicial).
O outro botão, denotado por **botão B**, controlará o sentido da transição de cores: horário ou anti-horário.

Ambos os botões poderão disparar interrupções externas. Porém, enquanto o botão A (velocidade) estará associado à interrupção externa INT0, configurada para responder a bordas de subida, o botão B (sentido)

estará associado a uma interrupção do tipo *pin-change*. Mais especificamente, escolhemos a PCINT1, gerada a partir do pino 0 da porta C, o qual está indicado no manual do microcontrolador como PCINT8.

Nesta atividade, vamos empregar uma solução baseada em *hardware* para o efeito de *bounce* nos dois botões. No caso, usaremos um circuito RC junto com um Schmitt *trigger* (7414) para eliminar as oscilações nos respectivos sinais.
Recomendação: para evitar riscos durante a montagem do circuito, utilizem capacitores de poliéster. O valor da capacitância pode ser de 100 nF, enquanto a resistência pode ser de 10 $k\Omega$.

Vamos, ademais, explorar os recursos oferecidos pela biblioteca **interrupt.h** para facilitar um pouco o tratamento de interrupções através de um programa desenvolvido em linguagem C. Assim, não será necessário configurar manualmente o vetor de interrupções.
Link: https://www.microchip.com/webdoc/AVRLibcReferenceManual/group__avr__interrupts.html

Para manter o LED RGB em um determinado estado durante certo intervalo de tempo, vamos utilizar a rotina de atraso desenvolvida em *Assembly* na atividade anterior. Para que o programa principal, escrito em linguagem C, "enxergue" a rotina preparada em *Assembly*, é preciso:

- Declarar o protótipo desta rotina no programa em C usando o qualificador **extern**, conforme o exemplo abaixo:

```
extern void atraso (short int q);
```

- Adicionar ao projeto em C no Atmel Studio o arquivo .s com o código em *Assembly* referente à rotina de atraso.
- Definir como global o símbolo correspondente à rotina de atraso no programa em *Assembly*. Por exemplo, se a rotina se chamada DELAY, então logo no início do código em *Assembly*, é preciso colocar a seguinte pseudo-instrução:

```
.global DELAY
```

Convenção de pinos

Por convenção, vamos empregar os pinos 0 a 2 da porta B (ou seja, PB0 a PB2, indicados no diagrama de pinos do Arduino UNO) para acionar os terminais correspondentes às cores vermelha (R), verde (G) e azul (B) do LED RGB, respectivamente.
O botão de controle da velocidade, por sua vez, deve ser conectado ao pino associado à interrupção externa INT0 (pino 2 da porta D).
Finalmente, como vamos utilizar a interrupção PCINT1, o botão que controla o sentido da transição de cores deve ser conectado ao pino 0 da porta C.

Materiais de consulta

Para esta atividade, é recomendado consultar o material sobre interrupções externas e *pin-change*, assim como o *link* com a descrição da biblioteca **interrupt.h**.
Para auxiliar na parte da montagem, é interessante rever o material sobre botões, principalmente a exposição da solução em *hardware* para *debounce*, e o material sobre LEDs RGB.

Instruções para a submissão do trabalho

1) Nos comentários do código-fonte (main.c), justifique as operações e os valores carregados em todos os registradores.
2) Crie um projeto chamado 'atividade5' (letras minúsculas, sem espaço) no Atmel Studio e, ao final da atividade, salve o diretório completo em um arquivo no formato .zip (Aviso: não use .tar.gz nem .rar), com nome 'seu_ra.zip' (Exemplo: 025304.zip).
3) Faça o *upload* da sua solução da atividade 5 no Google Classroom.