Atividade_01 - Revisão de Conceitos. Livro AVR e Arduino – Técnicas de Projeto
Capítulos: 1-Introdução e 2-O ATmega328

REGINALDO GREGÓRIO DE SOUZA NETO
2252813

Obs.: Deve ser entregue arquivo contendo as perguntas e respectivas respostas.

1. Atualmente, os microcontroladores estão presentes em quase todos os dispositivos eletrônicos controlados digitalmente. Cite três exemplos destes equipamentos encontrados nas casas, nos veículos e nos eletrônicos portáteis. Obs.: Três exemplos de cada categoria.

CASA: Micro-ondas, Ar condicionado e Geladeira.
VEÍCULOS: Controladora do painel, sistema de injeção eletrônica e alarme.
ELETRÔNICOS PORTÁTEIS: Smart-Watch, fones de ouvido e aparelho medidor de pressão.

2. Quanto à organização do barramento, existem duas arquiteturas predominantes para as CPUs dos microprocessadores, a arquitetura Von-Neumann e a arquitetura Harvard. Explique as características de cada uma delas.

A arquitetura de Von-Neumann se destaca pela característica de circulação de dados e instruções através de um único barramento. Devido à isso, dados e instruções não podem ser executados ao mesmo tempo, e para superar essa limitação, é necessária a busca antecipada de instruções e/ou com caches de instruções/dados.
A arquitetura de Harvard por sua vez, se caracterizada por dois barramentos internos, um de instruções e outro de dados. Deste modo, os dados e instruções podem ser acessados simultaneamente, o que torna essa arquitetura inerentemente mais rápida que a Von-Neumann.

3. Utilize a Figura 1.4 para explicar com um exemplo de código a diferença entre os Computadores com Conjunto Complexo de Instruções (CISC - Complex Instructions Set Computers) e Computadores com Conjunto Reduzido de Instruções (RISC – Reduced Instructions Set Computers).

O CISC consiste em instruções mais complexas, deste modo faz com que o processador gaste mais ciclos de clock para executar uma única instrução.

```
MULT 0,3
```

O RISC consiste em instruções mais simples, próximas ao que o processador realiza em cada ciclo de clock.

```
LOAD A,0
LOAD B,3
MULT A,B
STORE 0,A
```

Ambos os códigos descritos acima realizam uma operação de multiplicação, entretanto como pode ser observado o método RISC descreve as ações como um "passo a passo" do que o processador deve fazer. Para a arquitetura atual, como a CPU utiliza o mesmo barramento da memória com os registradores, isso pode ser útil caso seja necessário reutilizar os dados contidos nos registradores.

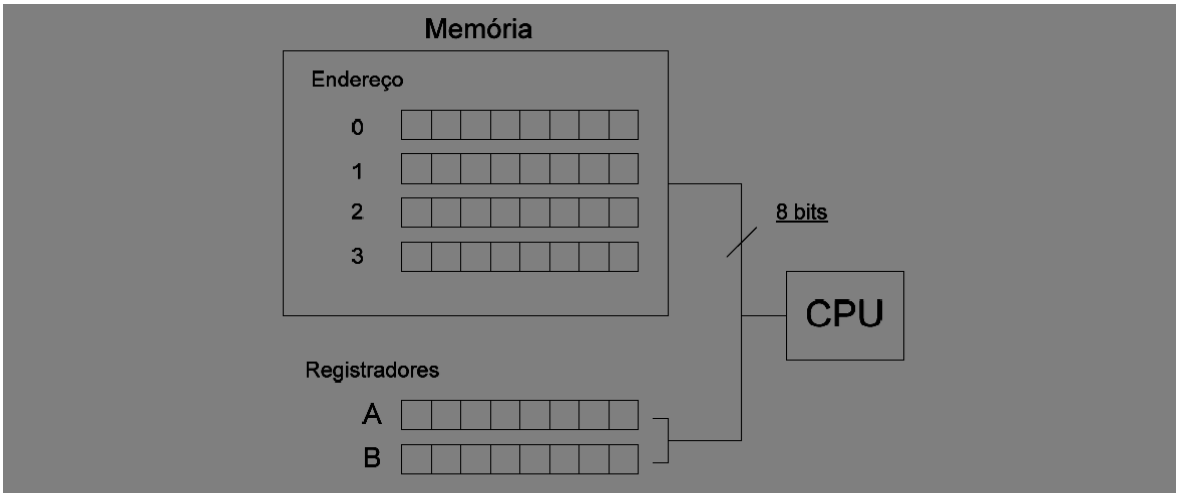


Fig. 1.4 – Diagrama esquemático para a comparação entre um microprocessador CISC e um RISC.

4. Defina o que é um microcontrolador e descreva as funcionalidades oferecidas por eles.

Um microcontrolador é o um sistema microprocessado com várias funcionalidades disponíveis em um único chip. Basicamente, um microcontrolador é um microprocessador com memórias de programa, de dados e RAM, temporizadores e circuitos de clock embutidos.
Dentre as funcionalidades encontradas nos microcontroladores é possível encontrar um gerador interno independente de clock; memória SRAM, EEPROM e flash; conversores analógicos-digitais (ADCs), conversores digitais-analógicos (DACs); vários 10 temporizadores/contadores; comparadores analógicos; saídas PWM; diferentes tipos de interface de comunicação, incluindo USB, USART, I2C, CAN, SPI, JTAG, Ethernet; relógio de tempo real; circuitos para gerenciamento de energia no chip; circuitos para o controle de inicialização (reset); alguns tipos de sensores; interface para LCD; e outros periféricos de acordo com o fabricante.

5. Descreva sobre 10 das principais características do microcontrolador ATmega328.

1 kbytes de memória EEPROM.
2 kbytes de memória SRAM.
Seção opcional para código de boot para programação In-System por boot loader6.
Bits de bloqueio para proteção contra a cópia do firmware.
Operação de até 20 MIPS a 20 MHz.
Multiplicação por hardware em 2 ciclos de clock 32 kbytes de memória de programa flash de auto programação In-System.
Microcontrolador de baixa potência, com arquitetura RISC avançada.
131 instruções, a maior parte executada em 1 ou 2 ciclos de clock (poucas em 3 ou 4 ciclos).
Ciclos de escrita e apagamento: memória flash 10 mil vezes, EEPROM 100 mil vezes.
32 registradores de trabalho de propósito geral (8 bits cada). Alguns trabalham em par para endereçamentos de 16 bits.

6. Quais são os 11 periféricos do microcontrolador ATmega328?

Interface serial USART.
Interface serial SPI Master/Slave.
Watchdog Timer com oscilador interno separado.
1 comparador analógico.
23 entradas e saídas (I/Os) programáveis.
Contador de tempo real (com um cristal externo de 32,768 kHz conta precisamente 1 s).
6 canais PWM. 6. 8 canais AD com resolução de 10 bits na versão TQFP (Thin profile plastic Quad Flat Package) e 6 canais na versão PDIP (Plastic Dual Inline Package).

Interface serial para dois fios orientada a byte (TWI), compatível com o protocolo I2C.
2 Temporizadores/Contadores de 8 bits com Prescaler separado, com modo de comparação.
1 Temporizador/Contador de 16 bits com Prescaler separado, com modo de comparação e captura.

7. O que é boot loader e onde está localizado na memória do microcontrolador Atmega328?

O Boot loader nada mais é do que um pequeno programa que pode ser escrito no início ou no final da memória de programa e serve para que o microcontrolador gerencie a gravação de sua memória. Para que isso ocorra, é necessário uma interface de comunicação externa com o software de desenvolvimento.

8. Explique a arquitetura Harvard empregada pelo ATmega328 mostrada no diagrama da Figura 2.1.
a) Barramento de dados.
b) Barramento de instruções.
c) Barramento de endereços.

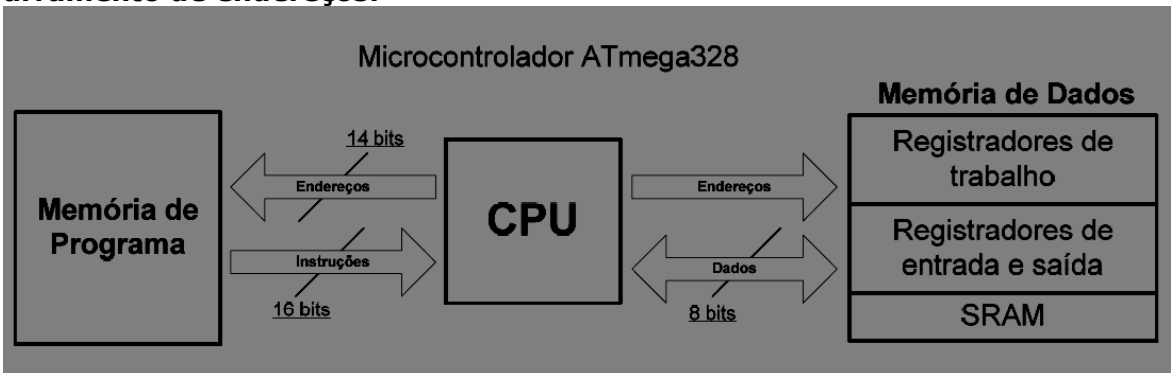


Fig. 2.1 – Diagrama esquemático da estrutura de um microcontrolador ATmega328.

O barramento de dados é de 8 bits, caracterizando o número de bits do microcontrolador. As instruções do ATmega são de 16 ou 32 bits (a maioria é de 16 bits). Assim, cada instrução consome 2 ou 4 bytes na memória de programa (um byte par e um ímpar). O acesso às posições de memória, dado pelo contador de programa (Program Counter - PC), é realizada de dois em dois bytes, começando sempre por uma posição par. Portanto, o barramento de endereços deve ser capaz de endereçar sempre posições pares da memória de programa. Logo, o bit menos significativo do barramento de endereços pode ser desprezado. Desta forma, para a memória de 32 kbytes do Atmega328 são necessários 14 bits de endereçamento (214 =16.384 endereços) e não 15 (215 = 32.768 endereços).

9. Utilizando o diagrama de tempo mostrado na Figura 2.4, explique a técnica de Pipeline empregada na arquitetura do ATmega328.

A arquitetura do AVR permite a busca e execução de instruções em paralelo devido ao emprego da técnica de Pipeline. Assim, o desempenho alcançado pode chegar a 1 MIPS por MHz. O diagrama de tempo da busca e execução de instruções, com referência ao clock da CPU, é apresentado na fig. 2.4. Observa-se que uma instrução é executada enquanto a próxima é lida. O ATmega emprega um pipeline de 2 estágios, dessa forma uma instrução é lida e decodificada dentro de um mesmo ciclo de clock, enquanto a instrução anterior é executada.

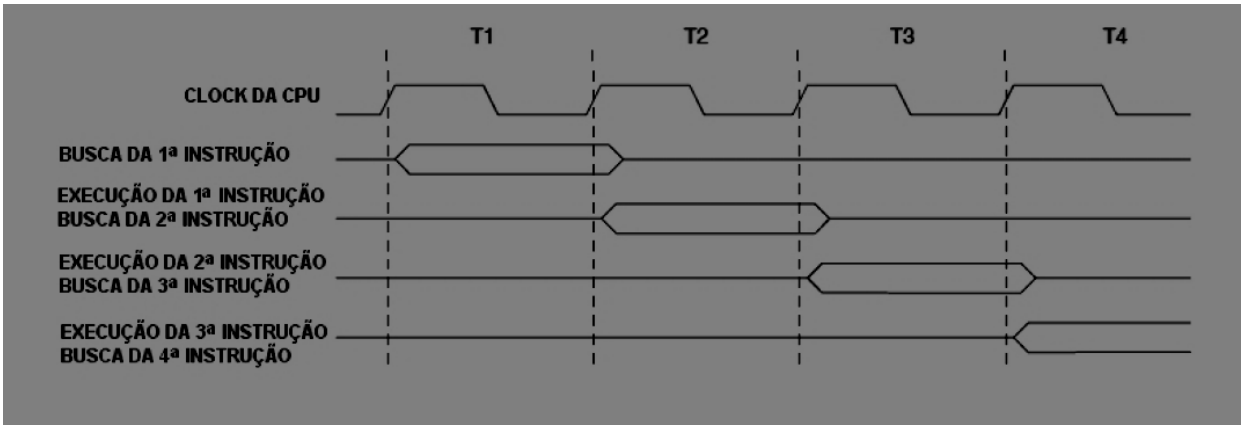


Fig. 2.4 – Diagrama de tempo para a busca e execução de instruções no ATmega.

10. Utilizando a Figura 2.5, explique a organização das memórias de dados e SRAM do ATmega328.

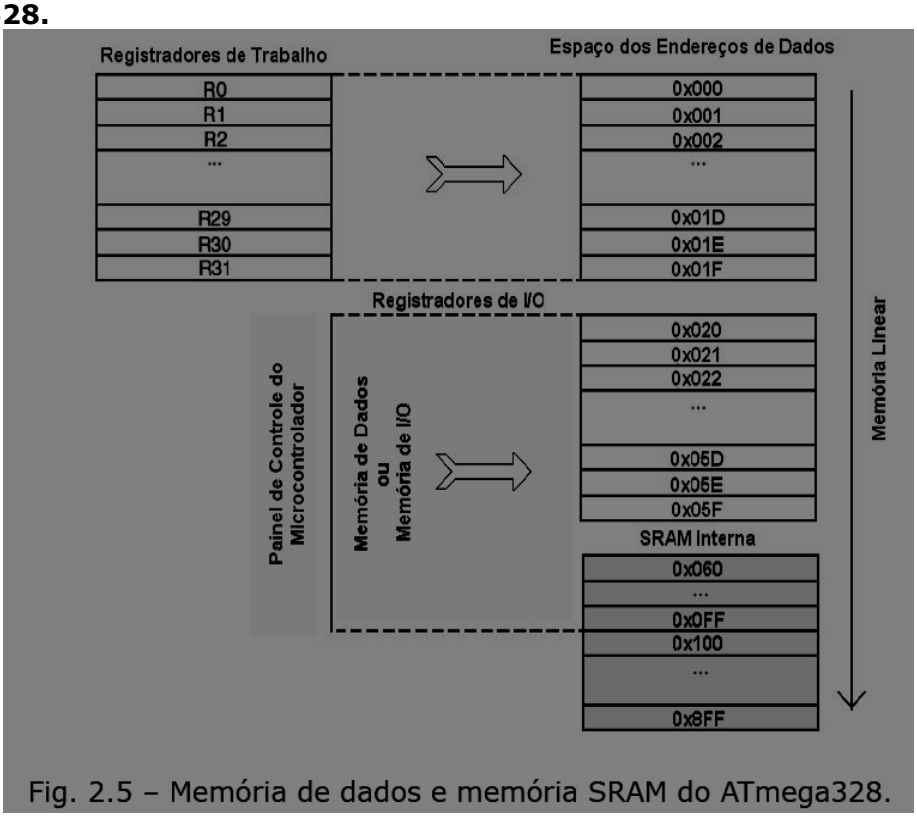


Fig. 2.5 – Memória de dados e memória SRAM do ATmega328.

A organização da memória é linear, começando no endereço 0 e indo até o endereço 2303 (0x8FF). Destas posições de memória, 32 pertencem aos registradores de uso geral (0x000 até 0x01F, fig. 2.2), 64 aos registradores de entrada e saída (0x020 até 0x05F) e 2048 bytes pertencem à memória SRAM (0x060 até 0x8FF), cujos 160 primeiros endereços são empregados para a extensão dos registradores de entrada e saída. Isso é necessário porque o número de periféricos no ATmega328 é superior ao que pode ser suportado pelos 64 registradores originais (dos primeiros ATmegas) e, também, para permitir o acréscimo de futuras funcionalidades. Desse modo, os engenheiros da Atmel utilizaram a SRAM7 para aumentar o número de registradores de I/O sem a necessidade de alterações profundas no projeto do chip.

11. Por que os registradores de I/O são chamados de "painel de controle" do microcontrolador?

Por que os registradores I/O dão acesso as entradas e as saídas, incluindo parte da memória e das funcionalidades do microcontrolador, pois possuem todas as informações referentes aos periféricos e ao processamento da CPU.

12. Explique o que é um PORT e como eles estão organizados no microcontrolador ATmega328.

O PORT é uma porta bidirecional de I/O de 8 bits com resistores internos de pullup selecionáveis para cada bit. E eles estão organizados por conjuntos (PORTB, PORTC, PORTD), possuindo 8 pinos cada, com exceção do PORTC.

13. O ATmega328 suporta várias opções de clock, identifique qual opção de clock e frequência utilizada na placa Arduino.

Cristal ou ressonador cerâmico externo, cristal de baixa frequência externo, sinal de clock externo e oscilador RC interno, Por padrão é utilizado a frequência de 1MHz, podendo chegar ao máximo de 8 MHz.

14. Descreva sobre as quatro fontes de RESET do ATmega328.

Power-on Reset: ocorre na energização enquanto a fonte de alimentação estiver abaixo da tensão limiar de power-on reset (VPOT).

Reset externo: ocorre quando o pino de reset é aterrado (0 V) por um determinado período de tempo.

Watchdog Reset: ocorre quando o watchdog está habilitado e o seu contador atinge o valor limite.

Brown-out Reset: ocorre quando a tensão de alimentação cair abaixo do valor definido para o brown-out reset (VBOT) e o seu detector estiver habilitado.

15. Utilizando a Figura 2.6, explique a organização da memória de programa do ATmega328.

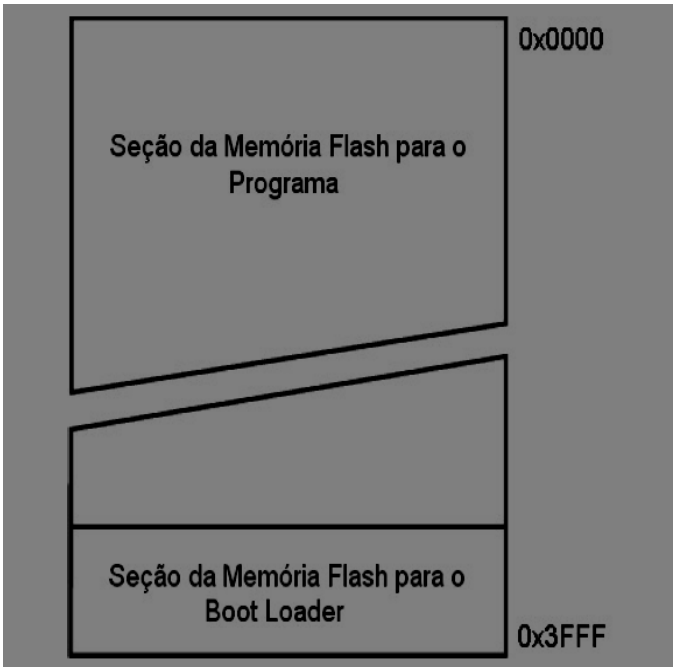


Fig. 2.6 – Organização da memória de programa.

Cada endereço da memória possui 2 bytes, pois as instruções do AVR são de 16 ou 32 bits. Dessa forma, a memória possui um total de 16384 endereços (de 0x0000 até 0x3FFF), correspondendo a 32 kbytes de memória. Existe uma seção específica para carregar o boot loader, que pode ou não ser utilizada para esse fim. A memória flash suporta, no mínimo, 10 mil ciclos de escrita e apagamento.

16. Utilizando a Figura 2.7, explique o funcionamento do Stack Pointer do ATmega328.

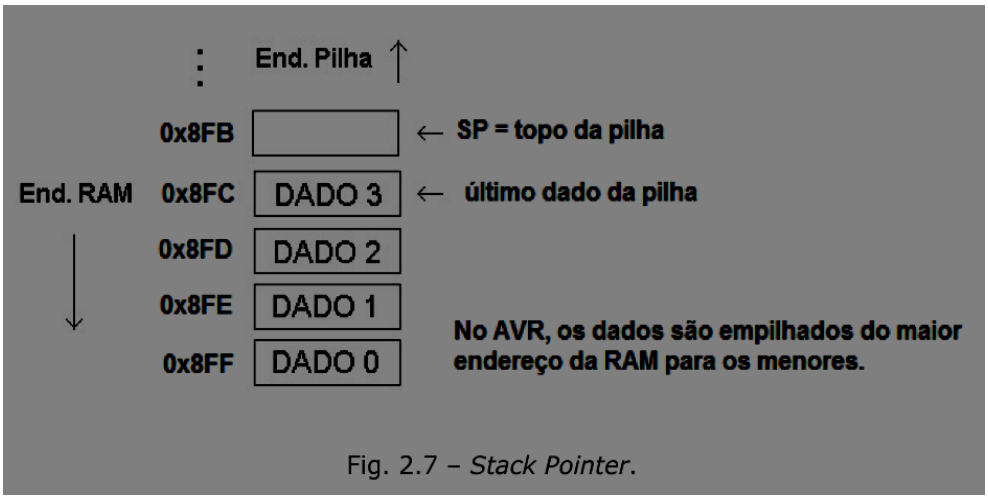


Fig. 2.7 – *Stack Pointer*.

O Stack Pointer (SP, ponteiro de pilha) é um registrador que armazena um endereço correspondente a uma posição da memória RAM, a qual é utilizada na forma de uma pilha para armazenagem temporária de dados: variáveis locais e endereços de retorno após chamadas de sub-rotinas e interrupções. Em resumo, o SP indica a posição onde um determinado dado foi armazenado na pilha alocada na RAM, esse conceito é ilustrado na Fig. 2.7.
O endereço do SP é pós-decrementado toda vez que um dado é colocado na pilha. Assim, a cada novo dado colocado na pilha, o endereço do SP apresenta um valor menor que o anterior. Da mesma forma, quando um dado é retirado da pilha, o endereço do SP é pré-incrementado, sempre apontando uma posição acima do último dado válido da pilha. Isso implica que o comando PUSH, que coloca um dado na pilha, diminui o valor do SP e o comando POP, que retira um dado da pilha, o incrementa.
Dado o funcionamento do SP, na sua inicialização ele deve apontar para o endereço final da RAM, no caso do ATmega328 o endereço é 0x8FF (seu valor default após a energização). Entretanto, existem microcontroladores da família ATmega que precisam ter o SP inicializado pelo programador.

Parte prática

Título: Conhecendo o Tinkercad para simulação de circuitos

Objetivos: Familiarização com o Tinkercad para a simulação de circuitos simples.

Nesta prática utilizaremos o Tinkercad para simular um circuito simples usando o microcontrolador Atmega328p, utilizado nas placas Arduino UNO.

Procedimentos:

1. Crie uma conta no Tinkercad, caso não possua (tinkercad.com).

2. Em seguida, vá para a aba circuits (https://www.tinkercad.com/circuits).

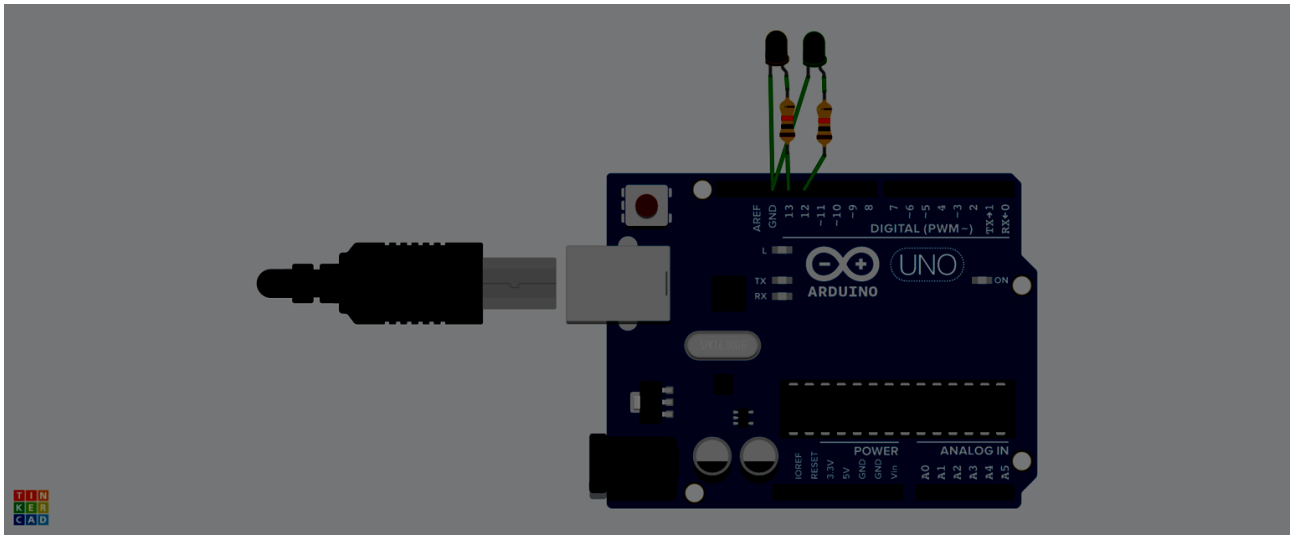
3. Você deve fazer um circuito capaz de piscar um led. Note que este projeto já está disponível (na aba Starters → Arduino).

4. Modifique o projeto de forma a provocar flashs intermitentes. O led deve ficar apagado por 500ms e aceso por apenas 50ms.

5. Adicione um segundo led que acende na sequencia do primeiro. Assim, a sequencia de ativação seria: LED1 (50ms), LED2(50ms), 450ms, LED1 (50ms), …

6. Use um led amarelo para o LED1 e verde para o LED2.

7. Cole o código fonte do microcontrolador ao final deste arquivo e inclua a imagem de seu design.




```
// C++ code
//REGINALDO GREGÓRIO DE SOUZA NETO
void setup()
{
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop()
{

  digitalWrite(13, HIGH);
  digitalWrite(12, HIGH);
  delay(50); // Wait for 50 millisecond(s)
  digitalWrite(13, LOW);
  digitalWrite(12, LOW);
  delay(450); // Wait for 450 millisecond(s)
}
```

Rúbrica:
Questões 01 a 11: 4% cada
Questões 12 a 16: 7% cada
Prática: Código coerente e funcional: 10%, design do circuito: 4%

Valor desta atividade na média: 0.5