EEPROM AVR

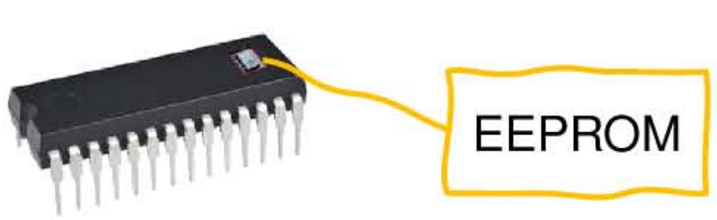
Electrically Erasable Programable Read Only Memory

© Prof. Engº esp Luiz Antonio Vargas Pinto
www.vargasp.com

Revisado 01/2021

Características

- A família Microchip, assim como a grande maioria dos microcontroladores, trazem em seu interior uma quantidade de memória EEPROM - o ATMEGA328p também possui.



- O ATMEGA328p possui 1024 endereços de EEPROM
  - ☒ De 0x000 a 0x3FF
- Armazena um byte em cada endereço

© Prof. Vargas

Sintaxe

```
EEPROM_Write(endereco,dado)
```

- Cuidado pois endereço é `int` e dado é `char`
  - ☒ **char** ocupa 1 byte e cabe de **0** a **255**
  - ☒ **int** ocupa 2 bytes e cabe de **0** a **65535**

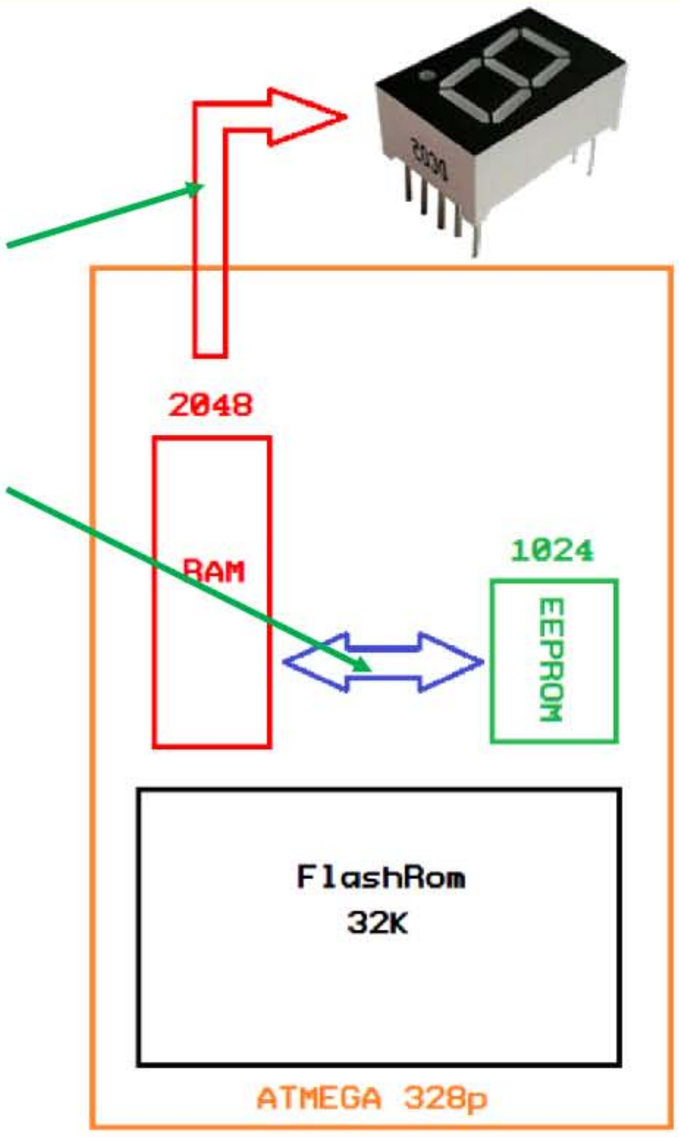
```
Destino = EEPROM_Read(endereco)
```

- Destino (**char** ou **int**)
- Endereço deve ser **int**

© Prof. Vargas

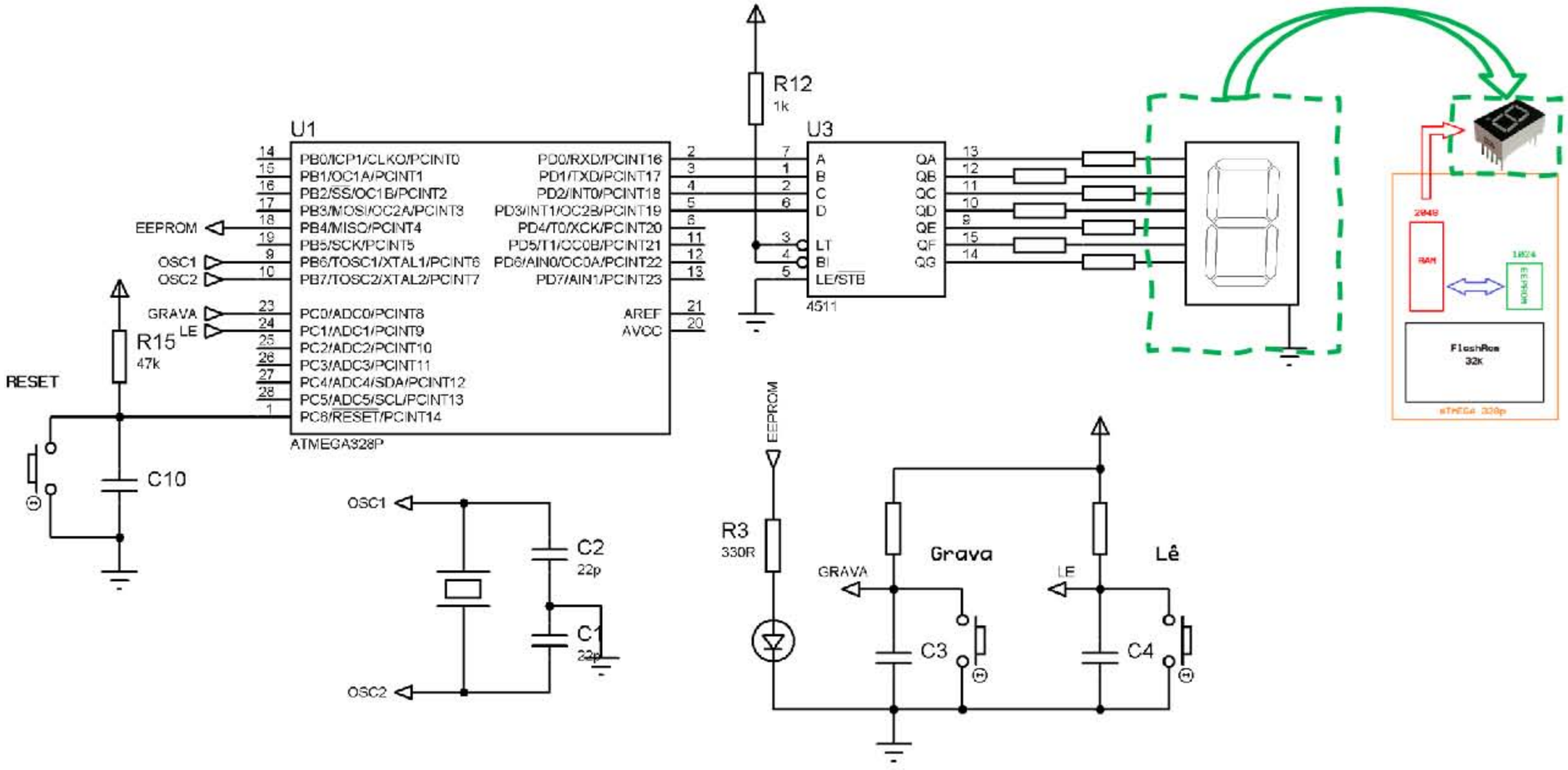
Proposta para teste da EEPROM

- O diagrama mostra que o display mostra apenas o conteúdo da RAM e que há uma comunicação entre a RAM e a EEPROM



© Prof. Vargas

Esquema de teste



© Prof. Vargas

Programa

```
/*-----------------------------------------------------------------------
  Programa : EEPROM1.c
  Autor    : Prof. Vargas
  Data     : 28/10/2019

  Descrição:
  Testa o uso da EEPROM Interna do AVRC que inicia em 0x000 e vai até 0x3FF.
  Este programa apresenta no display 7 segmentos o valor contido nos endereços
  do vetor de dados (de A[0]a A[9]).  Uma sequência de dados consecutivos de
  0 a 9 são gravados na EEPROM nos endereços 0x00 a 0x09 (coincidentemente).
    Após isto, a memória é preenchida com valores aletórios para que se possa
  ter a percepção da alteração ao se ler a EEPROM.  O botão LE, o AVR recolhe
  os dados da EEPROM e os recoloca na RAM. O botão GRAVA preenche a EEPROM com
  0s e 1s alternadamente, que podemos ler com o botão LE.
-----------------------------------------------------------------------*/
int endereco;
char k,dado,result;
char A[10] = {0,0,0,0,0,0,0,0,0,0};
```

© Prof. Vargas

```
void main()
{
    SREG.B7 = 0;     // Desabilita interrupção global
    PORTB = 0b00000000;
    PORTC = 0b00000000;
    PORTD = 0x00;           // 0 para o display

    DDRB  = 0b00111111;    // Port imita o Hardware
    DDRC  = 0b10111100;    // Port imita o Hardware
    DDRD  = 0b11111111;    // Port imita o Hardware

    //preenche EEPROM original sequencial
    for(endereco=0;endereco<10;endereco++)
    {
        EEPROM_Write(endereco,endereco);
    }
```

© Prof. Vargas

```
    //preenche dados visuais
    A[0] = 8;
    A[1] = 5;
    A[2] = 7;
40  A[3] = 0;
    A[4] = 2;
    A[5] = 6;
    A[6] = 4;
    A[7] = 1;
    A[8] = 3;
    A[9] = 9;
```

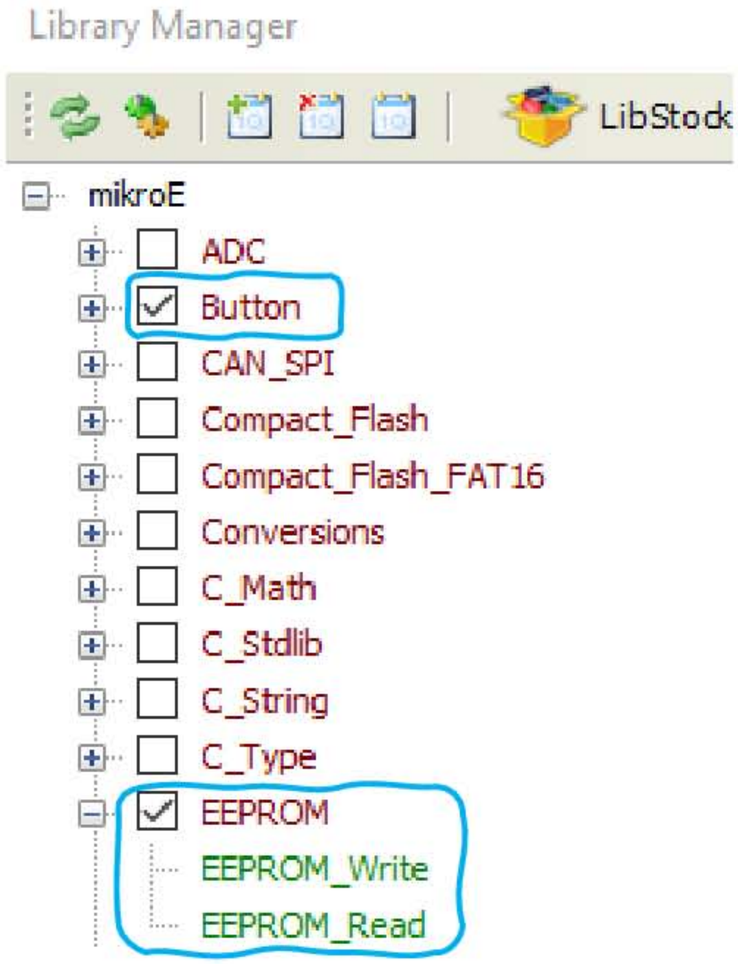
RAM

© Prof. Vargas

```
for(;;)    // Loop permanente
{
  for(k=0;k<10;k++)
  {
    PORTD = A[k];    //Apresenta dados
    if(Button(&PINC,0,1,0)) //GRAVA foi apertado
    {
      dado=0;
      for(endereco=0;endereco<10;endereco++)
      {
        EEPROM_Write(endereco,dado);
        dado=!dado;
      }
    }
    if(Button(&PINC,1,1,0)) //LE foi apertado
    {
      for(endereco=0;endereco<10;endereco++)
      {
        A[endereco] = EEPROM_Read(endereco);
      }
    }
    Delay_ms(500);
  }
}
```

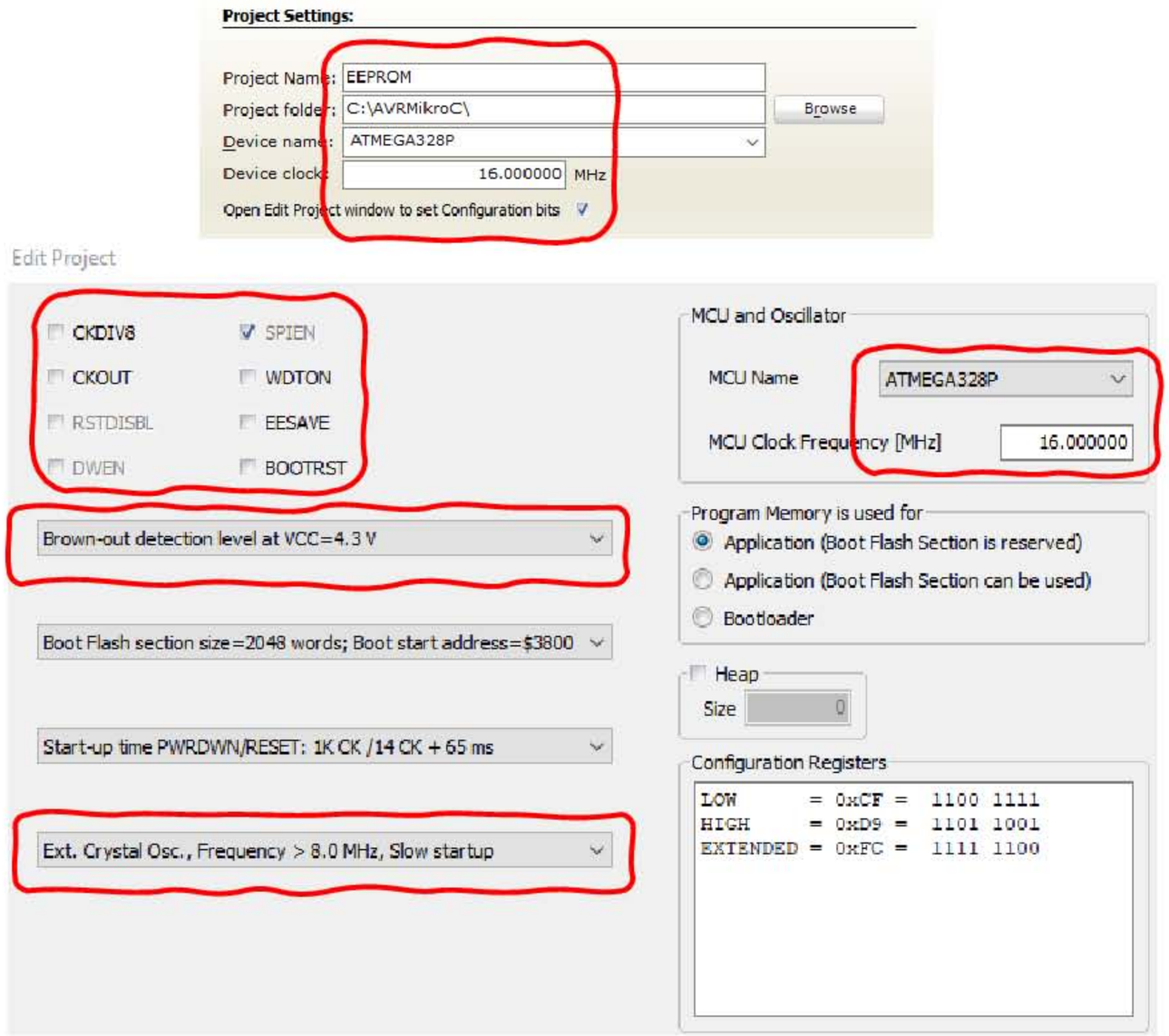
© Prof. Vargas

Usando a biblioteca do MikroC



© Prof. Vargas

Cuidados



© Prof. Vargas

© Prof. Vargas