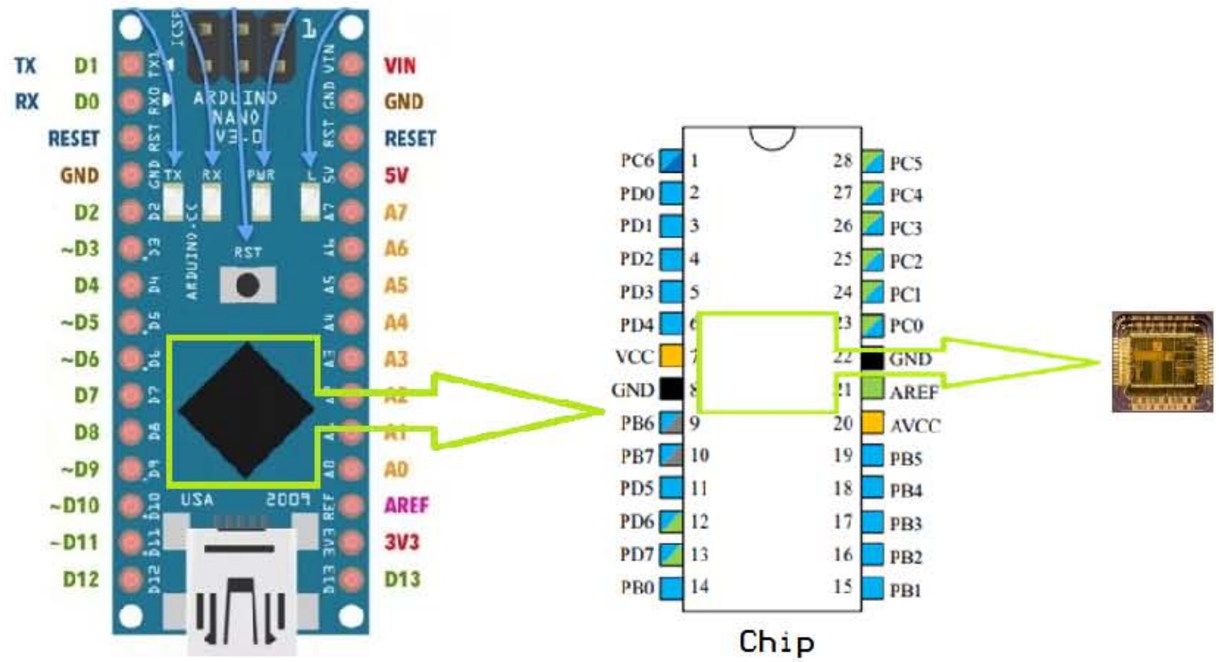
EEPROM NO NANO

Electrically Erasable Programable Read Only Memory



Plataforma

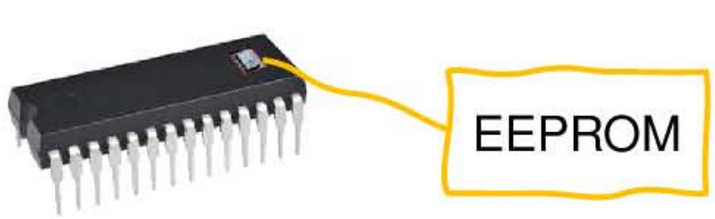
Chip

© 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 - como a base do Arduino é o ATMEGA328p, as características são as mesmas.



- O Arduino nano 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 `byte`
  - ☒ **byte** ocupa 1 byte e cabe de **0** a **255**
  - ☒ **int** ocupa 2 bytes e cabe de **0** a **65535**

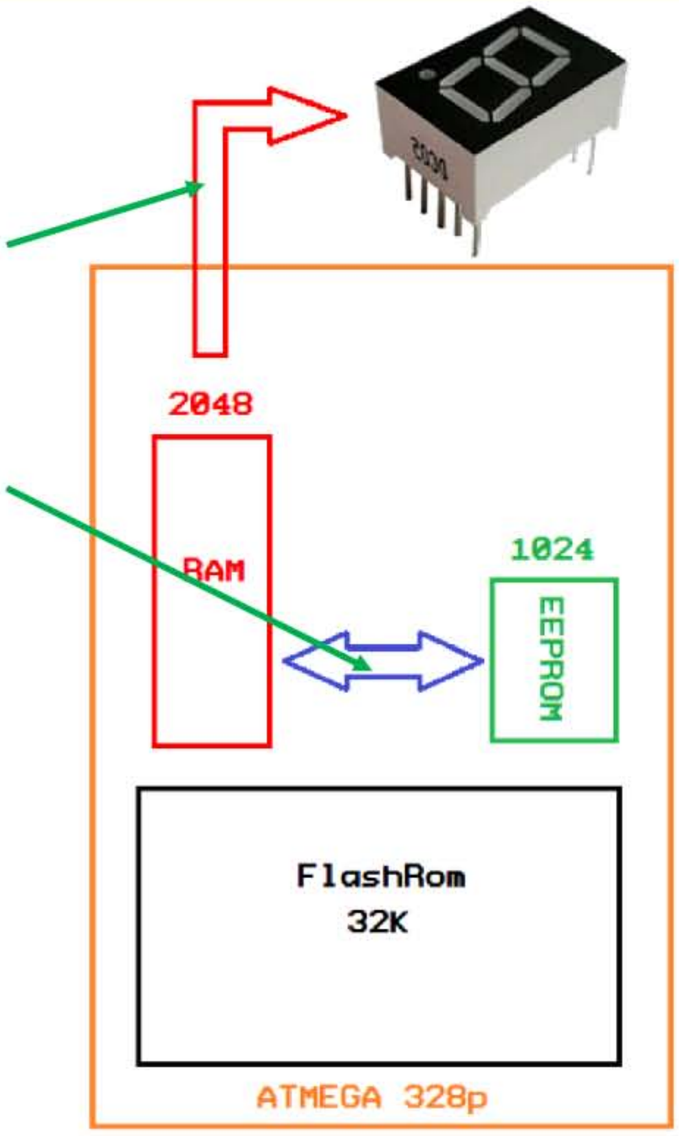
```
Destino = EEPROM.read(endereco)
```

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

© Prof. Vargas

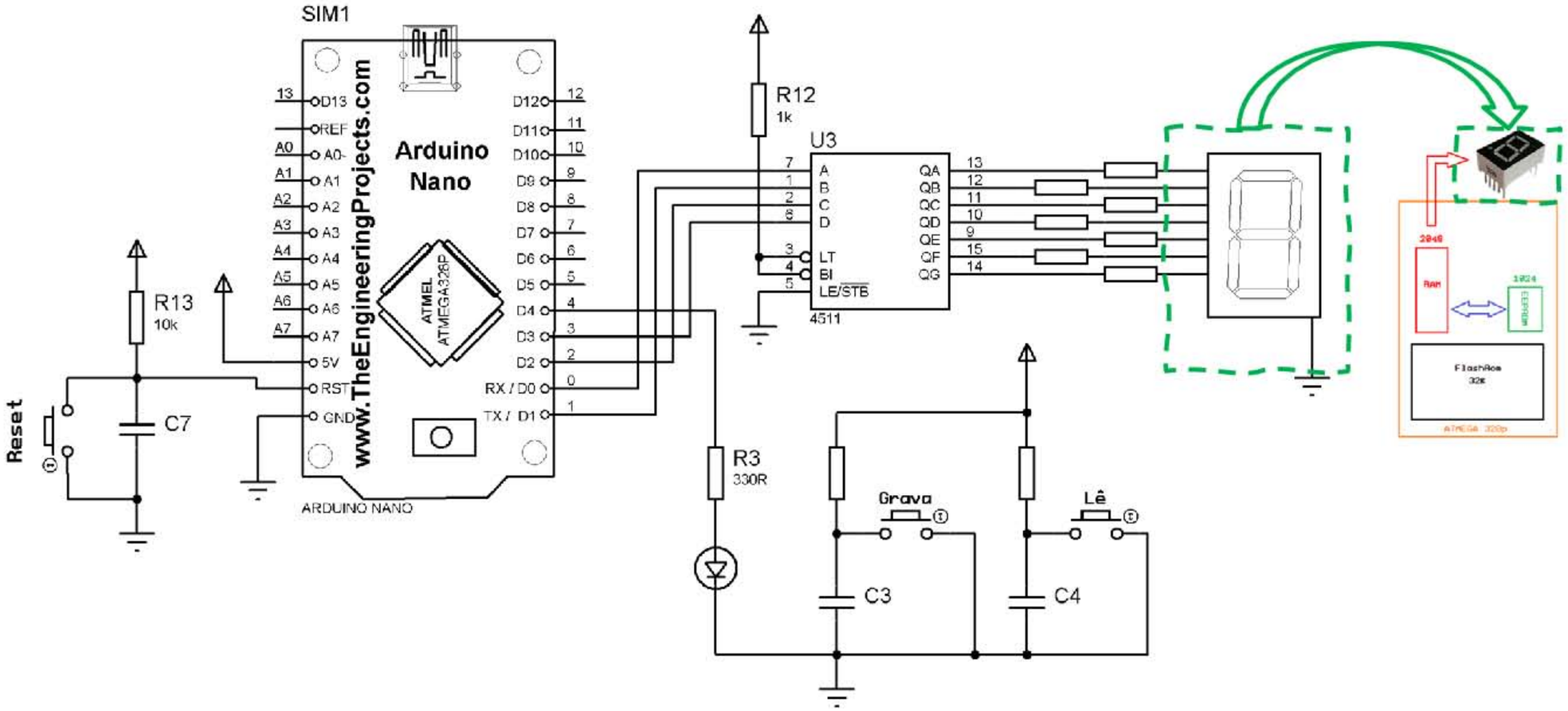
Proposta para teste da EEPROM

- 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

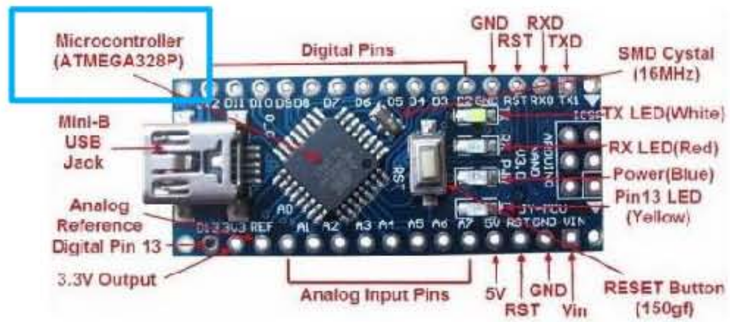
Programa

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

  Descrição:
  Testa o uso da EEPROM Interna do AVR 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 re-
  colhe 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.
  -----------------------------------------------------------------------------*/
#include <EEPROM.h>
const int Grava = 12;   //Batizamos o port de I/O digital 12 de Grava
const int Le = 11;      //Batizamos o port de I/O digital 11 de Le
int endereco,Botao;
byte k, dado, result;
char A[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

```

© Prof. Vargas

```
22 void setup()
23 {
24   pinMode(Grava, INPUT); //Definimos Grava como Output.
25   pinMode(Le, INPUT);    //Definimos Le como Input.
26   PORTD = 0x00;          // 0 para o display
27   DDRD  = 0b11111111;    // Port imita o Hardware
28
29   //preenche EEPROM original sequencial
30   for (endereco = 0; endereco < 10; endereco++)
31   {
32     EEPROM.write(endereco, endereco);
33   }
34
35   //preenche dados visuais
36   A[0] = 8;
37   A[1] = 5;
38   A[2] = 7;
39   A[3] = 0;
40   A[4] = 2;
41   A[5] = 6;
42   A[6] = 4;
43   A[7] = 1;
44   A[8] = 3;
45   A[9] = 9;
46 }
47
```

RAM



© Prof. Vargas

```
48 void loop()   // Loop permanente
49 {
50   for (k = 0; k < 10; k++)
51   {
52     PORTD = A[k];   //Apresenta dados
53     Botao = digitalRead(Grava);  //Recolhe estado do botão
54     if (Botao == LOW)    //GRAVA foi apertado
55     {
56       dado = 0;
57       for (endereco = 0; endereco < 10; endereco++)
58       {
59         EEPROM.write(endereco, dado);
60         dado = !dado;
61       }
62     }
63     Botao = digitalRead(Le);  //Recolhe estado do botão
64     if (Botao == LOW)    //LE foi apertado
65     {
66       for (endereco = 0; endereco < 10; endereco++)
67       {
68         A[endereco] = EEPROM.read(endereco);
69       }
70     }
71     delay(500);
72   }
73 }
```

© Prof. Vargas

© Prof. Vargas