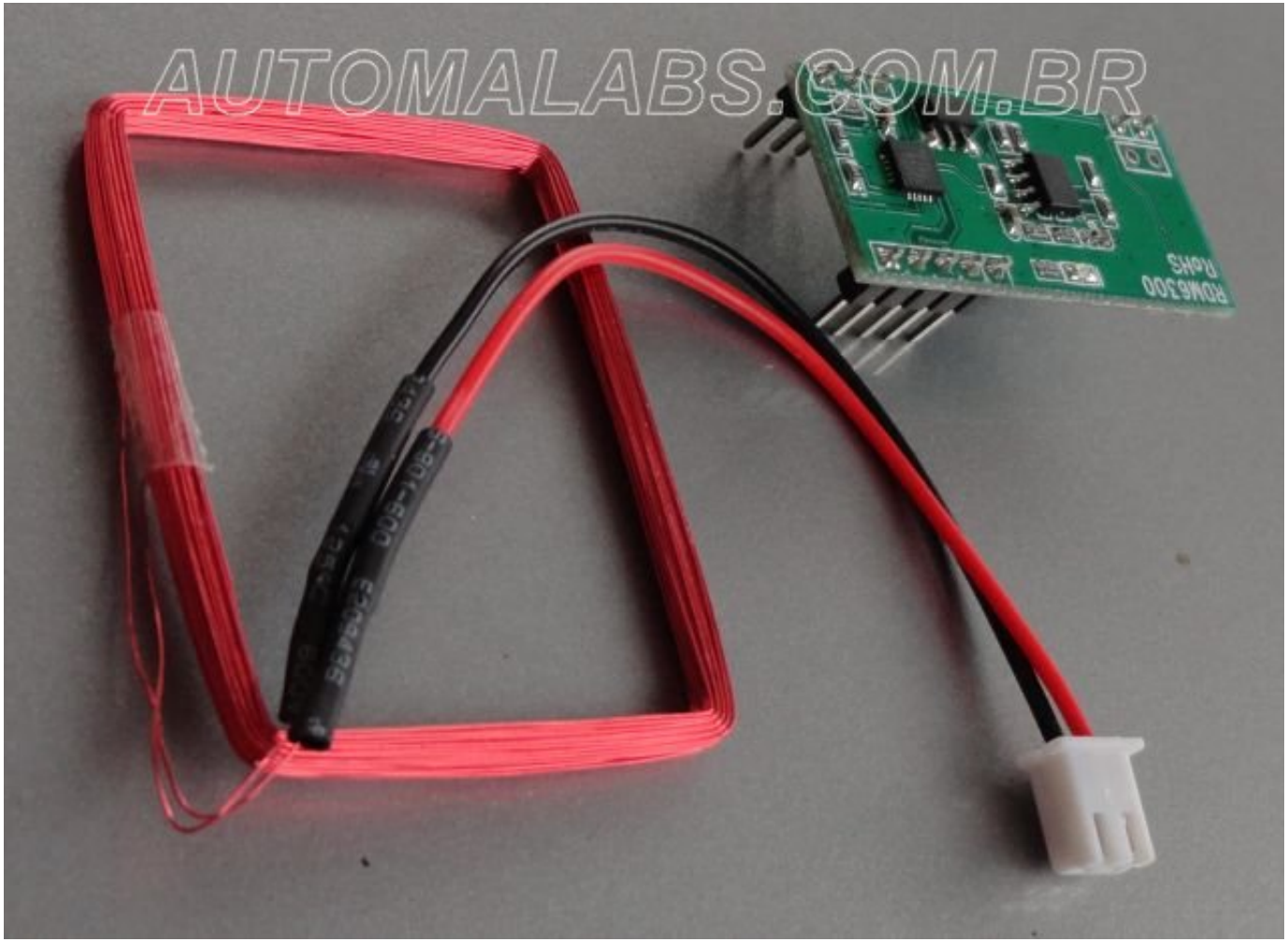
AUTOMALABS

Ferramentas para automação – Em Recife
RSS
‹ Prev Next ›
mar31

Módulo RFID 125KHz com saída UART – RDM6300

by Jefferson on 31 de março de 2013 at 21:35
Posted In: Sem categoria

Com este módulo você poderá ler tags RFID de 125khz no arduino, ou em qualquer computador, desde que use também um adaptador serial TTL-USB.

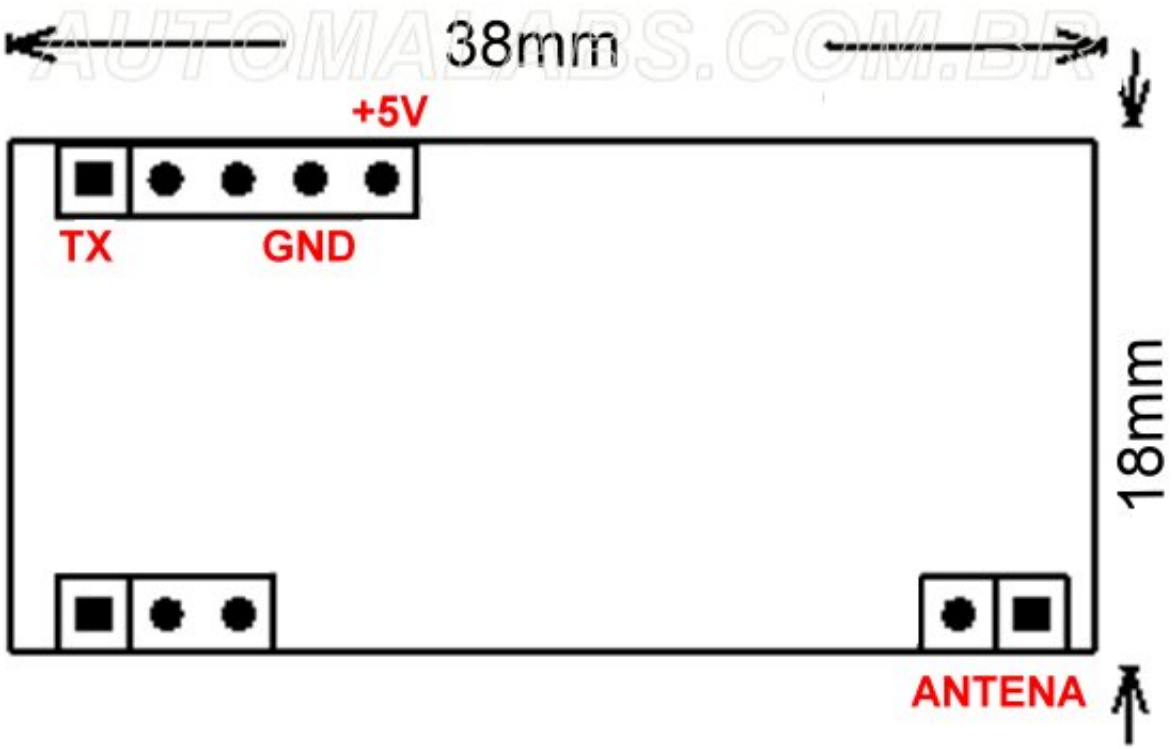

AUTOMALABS.COM.BR
RDM6300

Não deixe a quantidade de pinos intimidar você. A conexão mínima só requer três fios:

- +5V
- GND
- TX (para o Arduino, PC, etc)

Nota: O módulo tem um pino RX, mas seu propósito não é esclarecido.

Diagrama de conexão, com o módulo visto por baixo:



O módulo tem um microcontrolador próprio, que se encarrega de todo o trabalho de decodificação RFID, já entregando na saída o número da tag (chaveiro, cartão, etc).

A saída é serial UART TTL padrão (9600bps, 8,N,1). Para usar com qualquer Arduino, nenhum hardware extra é necessário, porque basta ligar o TX do módulo ao RX do Arduino. Caso você use um Arduino que só tem uma porta serial como o Duemilanove e o Uno, pode usar a biblioteca NewsoftSerial para criar uma porta "virtual" em qualquer outro pino e assim liberar a porta serial padrão para outras tarefas.

Para usar com um computador, é preciso usar um conversor USB-serial TTL ou RS232 – TTL.

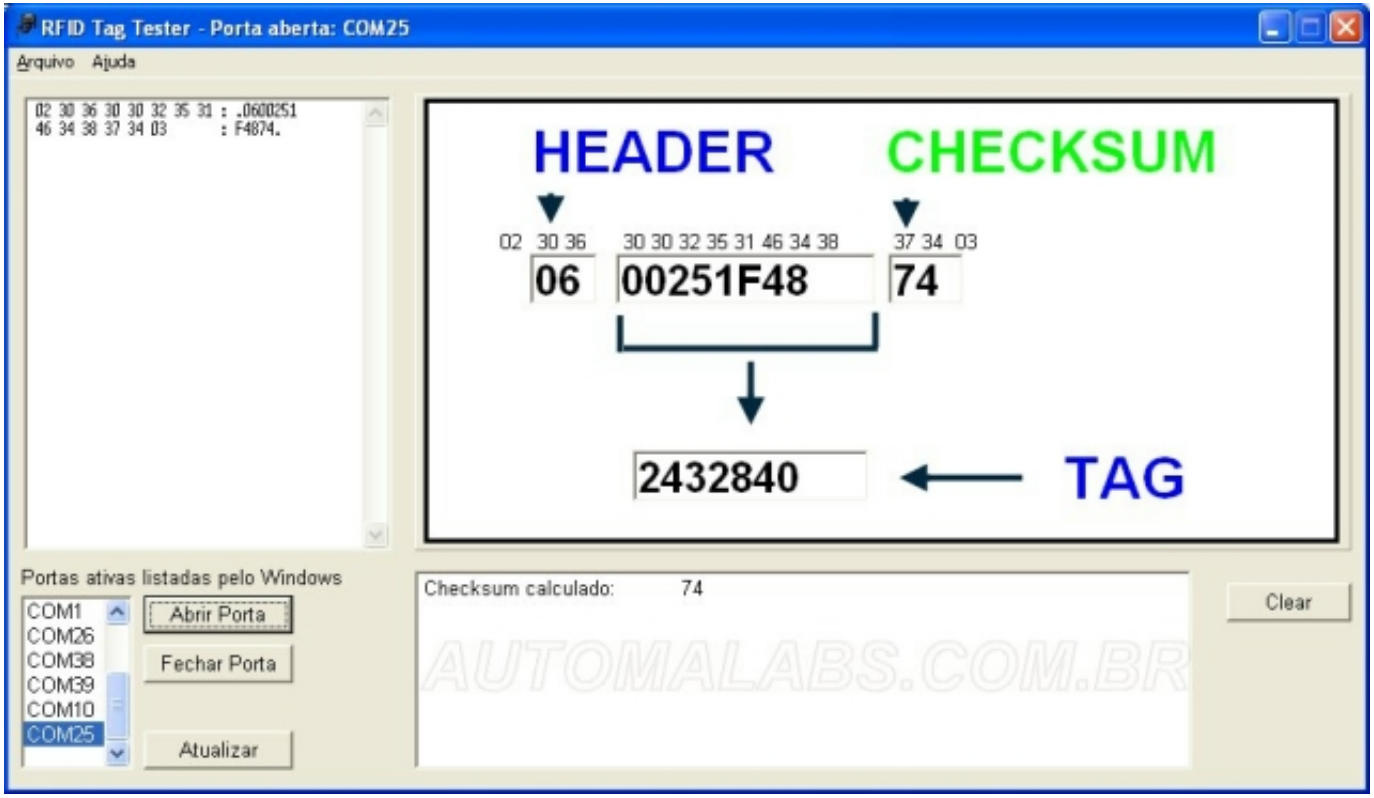
Alcance da antena: no máximo 3cm.

Proteção contra inversão de polaridade por diodo em série.

O número da tag é transmitido em um formato codificado que é explicado no datasheet em anexo, que é o único oferecido pelo fabricante. Eu o acho um bocado confuso, por isso escrevi um programa em Delphi, também em anexo, que ajudará você a entender como o número transmitido se

relaciona com o real, escrito na tag. Porém em boa parte das aplicações o número escrito na tag é irrelevante, pois o que importa é que o usuário apresente uma tag que está registrada no sistema. O número impresso só se torna importante quando você quer testar a tag ou correlacionar o uso das tags com um usuário específico.

Exemplo da tela do programa:



Para testar com um arduino você pode usar o ótimo exemplo de Mario Boehmer, que reproduzo abaixo ligeiramente traduzido (ligue o TX ao pino D6 do arduino):

Atenção: Requer a biblioteca NewsoftSerial e para funcionar no Arduino 1.x requer modificações (testei no 0.22).

```
1  //http://marioboehmer.blogspot.com.br/2011/01/rfid-with-arduino.html
2
3  #include "NewSoftSerial.h"
4  #define stx 2
5  #define etx 3
```

```
6
7   NewSoftSerial mySerial(6, 7);
8   int counter;
9   byte data[14];
10  byte hexBlock1,hexBlock2,hexBlock3,hexBlock4,hexBlock5;
11  byte hexCalculatedChecksum,hexChecksum;
12
13  void setup() {
14  Serial.begin(9600);
15  mySerial.begin(9600);
16  }
17
18  void loop() {
19  if (mySerial.available() > 0) {
20  data[counter] = mySerial.read();
21  counter++;
22  if(counter > 13) {
23  //we read the whole message, so reset counter
24  counter = 0;
25  //check if start of text and end of text is correct
26  if(data[0] == stx && data[13] == etx) {
27  Serial.println("Caracteres STX e ETX corretamente recebidos");
28  Serial.print("ID: ");
29  //show ID
30  for(int x = 1; x < 11; x++) {
31  Serial.print(data[x], BYTE);
32  }
33  Serial.println("");
34  Serial.print("Checksum: ");
35  //show checksum
36  Serial.print(data[11], BYTE);
37  Serial.println(data[12], BYTE);
38
39  //Hex ID blocks. Two transmitted Bytes form one Hex ID block.
40  //Hex ID blocks:      6   2  |  E   3  |  0   8  |  6   C  |  E   D
41  //Transmitted Bytes: 36H 32H | 45H 33H | 30H 38H | 36H 43H | 45H 44H
42  hexBlock1 = AsciiCharToNum(data[1])*16 + AsciiCharToNum(data[2]);
43  hexBlock2 = AsciiCharToNum(data[3])*16 + AsciiCharToNum(data[4]);
44  hexBlock3 = AsciiCharToNum(data[5])*16 + AsciiCharToNum(data[6]);
45  hexBlock4 = AsciiCharToNum(data[7])*16 + AsciiCharToNum(data[8]);
```

```
46  hexBlock5 = AsciiCharToNum(data[9])*16 + AsciiCharToNum(data[10]);
47
48  //Transmitted checksum.
49  hexChecksum = AsciiCharToNum(data[11])*16 + AsciiCharToNum(data[12]);
50
51  //XOR algorithm to calculate checksum of ID blocks.
52  hexCalculatedChecksum = hexBlock1 ^ hexBlock2 ^ hexBlock3 ^ hexBlock4 ^ hexBlock5;
53  if ( hexCalculatedChecksum == hexChecksum )
54  {
55  Serial.println("Checksum calculado confere com checksum transmitido.");
56  }
57  else {
58  Serial.println("Checksum calculado NAO confere com checksum transmitido. Dados corrompidos!");
59  }
60  }
61  }
62  }
63  }
64
65  uint8_t AsciiCharToNum(byte data) {
66  //First substract 48 to convert the char representation
67  //of a number to an actual number.
68  data -= '0';
69  //If it is greater than 9, we have a Hex character A-F.
70  //Substract 7 to get the numeral representation.
71  if (data > 9)
72  data -= 7;
73  return data;
74  }
```

RFID RDM6300 Tag Tester
RFID_RDM6300_Tag_Tester.exe
Version: 0.1 beta
1.1 MiB
63 Downloads
Detalhes…

RFID-RDM630-Spec
RFID-RDM630-Spec.pdf
199.0 KiB
50 Downloads
Detalhes...

└ Tags: Meios de Entrada

Comment ¬

O seu endereço de email não será publicado Campos obrigatórios são marcados *

NAME — Get a Gravatar

EMAIL

Website URL

NOTE - You can use these HTML tags and attributes:

```
<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime="">
<em> <i> <q cite=""> <strike> <strong>
```

Post Comment

Tópicos recentes

- Tenho peças para os projetos do livro Arduino Basico de Michael Mcroberts.
- Mudança na política de participação
- Módulo sensor de chuva / umidade com saídas digital e analógica
- Kit de motor cc com caixa de redução + roda para smartcar
- Sensores de pressão Freescale MPX5010 (1m) e MPX5050 (5m)

Meta

- Login
- Posts RSS
- RSS dos comentários
- WordPress.org

Categorias

- Sem categoria

Search... »